TortoiseSVN

Odjemalec za Subversion v operacijskem sistemu Windows

Version 1.10

Stefan Küng
Lübbe Onken
Simon Large

TortoiseSVN: Odjemalec za Subversion v operacijskem sistemu Windows: Version 1.10

od Stefan Küng, Lübbe Onken, in Simon Large
Prevod: Matjaž Čepon (matjaz.cepon@gmail.com)

Publication date 2018/03/17 15:14:17 (r28148)

Kazalo

Seznam slik

Seznam tabel

Predgovor

Nadzor različic je umetnost vodenja sprememb informacij. Že dolgo je orodje izjemnega pomena za programerje, ki tipično porabijo veliko časa za male spremembe na programski opremi, naslednji dan pa nekatere izmed njih razveljavijo ali preverijo. Zamislite si ekipo razvijalcev programske opreme, ki dela sočasno na istih datotekah in videli boste, zakaj je potreben dober sistem za *vodenje potencialnega nereda*.

1. Kaj je TortoiseSVN?

TortoiseSVN is a free open-source Windows client for the *Apache™ Subversion®* version control system. That is, TortoiseSVN manages files and directories over time. Files are stored in a central *repository*. The repository is much like an ordinary file server, except that it remembers every change ever made to your files and directories. This allows you to recover older versions of your files and examine the history of how and when your data changed, and who changed it. This is why many people think of Subversion and version control systems in general as a sort of “time machine”.

Nekateri sistemi za nadroz različic so posebej prikrojeni za urejanje strukture izvorne kode in imajo veliko zmožnosti, ki so specifične za razvoj programske opreme - na primer razumevanje programskih jezikov ali dodatna orodja za njhovo gradnjo. To so sistemi SCM (software configuration management). Subversion ni tak sistem; je splošen sistem, ki omogoča urejanje *katerokoli* zbirke datotek, vključno z datotekami izvorne kode.

2. Značilnosti TortoiseSVN

Zakaj je TortoiseSVN tako dober odjemalec za Subversion? Tukaj je kratek seznam njegovih značilnosti.

Integracija z lupino

TortoiseSVN integrates seamlessly into the Windows shell (i.e. the explorer). This means you can keep working with the tools you're already familiar with. And you do not have to change into a different application each time you need the functions of version control.

And you are not limited to using the Windows Explorer; TortoiseSVN's context menus work in many other file managers, and also in the File/Open dialog which is common to most standard Windows applications. You should, however, bear in mind that TortoiseSVN is intentionally developed as an extension for the Windows Explorer. Thus it is possible that in other applications the integration is not as complete and e.g. the icon overlays may not be shown.

Prekrivne ikone

Stanje vsake datoteke pod nadzorom različic nakazuje majhna prekrivna ikona. Tako lahko hitro vidite, kakšno je stanje vaše delovne kopije.

Graphical User Interface

When you list the changes to a file or folder, you can click on a revision to see the comments for that commit. You can also see a list of changed files - just double click on a file to see exactly what changed.

The commit dialog lists all the items that will be included in a commit, and each item has a checkbox so you can choose which items you want to include. Unversioned files can also be listed, in case you forgot to add that new file.

Preprost dostop do ukazov sistema Subversion

Vsi ukazi sistema Subversion so na razpolago v kontekstnem meniju Raziskovalca. TortoiseSVN doda tja svoj lasten pomeni.

Ker je TortoiseSVN odjemalec za Subversion, vam bomo prikazali tudi nekaj značilnosti samega sistema Subversion:

Vodenje različic map
: CVS upravlja le zgodovino posameznih datotek, Subversion pa ima "virtualni" datotečni sistem pod nadzorom različic, ki upravlja spremembe na celotnem drevesu map. Pod nadzorom so datoteke *in* mape. Posledica tega je, da imamo na strani odjemalca na razpolago ukaza **premakni** in **kopiraj**, ki delujeta na datotekah in mapah.

Atomične objave
: Objava zapiše vse spremembe v skladišče ali pa jih sploh ne zapiše. To omogoča razvijalcem, da sestavijo in objavijo spremembe kot logične celote.

Metapodatki pod nadzorom različic
: Vsaka datoteka in mapa ima prirejeno nevidno množico "lastnosti". Lahko si izmislite kakršen koli par ključ/vrednost. Lastnosti so pod nadzorom različic, prav tako kot vsebina datoteke.

Izbira plasti omrežja
: Subversion je vpeljal abstrakten koncept dostopa do skladišča, kar uporabnikom omogoča, da izdelajo nove omrežne mehanizme. Subversionov "napredni" mrežni strežnik je modul za spletni strežnik Apache, ki govori narečje protokola HTTP, imenovano WebDAV/DeltaV. To daje sistemu Subversion veliko prednosti s stališča stabilnosti in povezovanja, prinaša pa še številne dodatne zmožnosti, n. pr.: avtentikacijo, avtorizacijo, *wire compression* in brskanje po skladišču. Na razpolago pa je tudi manjši, samostojen strežnik za Subversion. Strežnik se pogovarja po prilagojenem protokolu, ki se ga preprosto preusmeri preko ssh.

Konsistentno upravljanje s podatki
: Subversion zapisuje razlike med datotekami z dvojiškim algoritmom za razlikovanje, ki deluje tako na tekstovnih (uporabniku berljivih) kot na dvojiških (uporabniku neberljivih) datotekah. Datoteke obeh tipov so enako stisnjene in shranjene v skladišču, razlike pa se prenašajo v obeh smereh po mreži.

Učinkovita uporaba vej in oznak
: Cena vej in oznak ni nujno sorazmerna z velikostjo projekta. Subversion ustvari veje in oznake tako, da projekt skopira z uporabo mehanizma, podobnega simbolnim povezavam. Tako ti operaciji trajata zelo kratek (konstanten) čas in zavzameta zelo malo prostora v skladišču.

3. License

TortoiseSVN is an Open Source project developed under the GNU General Public License (GPL). It is free to download and free to use, either personally or commercially, on any number of PCs.

Although most people just download the installer, you also have full read access to the source code of this program. You can browse it on this link https://sourceforge.net/p/tortoisesvn/code/HEAD/tree/. The current development line is located under `/trunk/`, and the released versions are located under `/tags/`.

4. Development

Both TortoiseSVN and Subversion are developed by a community of people who are working on those projects. They come from different countries all over the world, working together to create great software.

4.1. Zgodovina TortoiseSVN

In 2002, Tim Kemp found that Subversion was a very good version control system, but it lacked a good GUI client. The idea for a Subversion client as a Windows shell integration was inspired by the similar client for CVS named TortoiseCVS. Tim studied the source code of TortoiseCVS and used it as a base for TortoiseSVN. He then started the project, registered the domain `tortoisesvn.org` and put the source code online.

Around that time, Stefan Küng was looking for a good and free version control system and found Subversion and the source for TortoiseSVN. Since TortoiseSVN was still not ready for use, he joined the project and started programming. He soon rewrote most of the existing code and started adding commands and features, up to a point where nothing of the original code remained.

As Subversion became more stable it attracted more and more users who also started using TortoiseSVN as their Subversion client. The user base grew quickly (and is still growing every day). That's when Lübbe Onken offered to help out with some nice icons and a logo for TortoiseSVN. He now takes care of the website and manages the many translations.

With time, other version control systems all got their own Tortoise client which caused a problem with the icon overlays in Explorer: the number of such overlays is limited and even one Tortoise client can easily exceed that limit. That's when Stefan Küng implemented the TortoiseOverlays component which allows all Tortoise clients to use the same icon overlays. Now all open source Tortoise clients and even some non-Tortoise clients use that shared component.

4.2. Zasluge

Tim Kemp
: for starting the TortoiseSVN project

Stefan Küng
: for the hard work to get TortoiseSVN to what it is now, and his leadership of the project

Lübbe Onken
: za čudovite ikone, logotip, iskanje napak, prevode in urejanje prevodov

Simon Large
: for maintaining the documentation

Stefan Fuhrmann
: for the log cache and revision graph

The Subversion Book
: za izvrsten uvod v sistem Subversion in poglavje številka dve, ki smo ga uporabili v tej knjigi

The Tigris Style project
: za nekatere stile, ki smo jih uporabili v tej dokumentaciji

Zunanji sodelavci
: for the patches, bug reports and new ideas, and for helping others by answering questions on our mailing list

Naši donatorji
: za številne ure užitkov ob poslušanju glasbe, ki so nam jo podarili

5. Vodnik po knjigi

This book is written for computer-literate folk who want to use Subversion to manage their data, but prefer to use a GUI client rather than a command line client. TortoiseSVN is a windows shell extension and it is assumed that the user is familiar with the windows explorer and how to use it.

This Predgovor explains what TortoiseSVN is, a little about the TortoiseSVN project and the community of people who work on it, and the licensing conditions for using it and distributing it.

The Poglavje 1, *Kako začeti* explains how to install TortoiseSVN on your PC, and how to start using it straight away.

Poglavje Poglavje 2, *Basic Version-Control Concepts* poda nekaj osnov o sistemu za nadzor različic *Subversion*, ki je podlaga za TortoiseSVN. Ta del je izposojen iz dokumentacije Subversion in ponazori različne pristope k vodenju različic datotek in delovanje sistema Subversion.

The chapter on Poglavje 3, *Skladišče* explains how to set up a local repository, which is useful for testing Subversion and TortoiseSVN using a single PC. It also explains a bit about repository administration which is also relevant to repositories located on a server.

Poglavje 4, *Dnevna uporaba* je najpomembnejše poglavje, saj so v njem razložene vse glavne zmožnosti programa TortoiseSVN in njihova uporaba. Napisano je kot vodnik. Začne se z prevzemom delovne kopije, njenim spreminjanjem, nadaljuje pa z objavo narejenih sprememb. V nadaljevanju so razložene zahtevnejše tematike.

Poglavje 6, *Program SubWCRev* je zunanji program, ki je vključen v paket TortoiseSVN. Z njegovo pomočjo lahko izluščimo informacije iz delovne kopije in jih zapišemo v datoteko. To je uporabno, če želimo v projekt vključiti informacijo o številki revizije.

Poglavje Dodatek B, *Kako naredim...* poda odgovore na nekatera pogosta vprašanje o nalogah, ki niso opisana v ostalih poglavjih.

Poglavje Dodatek D, *Avtomatizacija TortoiseSVN* ponazori, kako lahko kličemo pogovorna okna TortoiseSVN iz ukazne vrstice. To je primerno za pisanje ukaznih datotek, v katerih še vedno želimo uporabnikove odzive.

Poglavje Dodatek E, *Ustrezni ukazi v odjemalcu za ukazno vrstico* poda seznam ukazov TortoiseSVN in njihovih sorodnih ukazov v odjemalcu Subversion za ukazno vrstico (`svn.exe`).

6. Uporabljena terminologija

Da bi olajšali branje knjige, so imena vseh oken in menijev TortoiseSVN označena z drugo barvo, npr. Dnevnik.

Izbira v meniju je prikazana s puščico. TortoiseSVN → Pokaži dnevnik pomeni: iz kontekstnega menija *TortoiseSVN* izberite *Pokaži dnevnik*.

V primeru, ko se kontekstni meni pojavi znotraj enega od pogovornih oken sistema TortoiseSVN, je to prikazano takole: Kontekstni meni → Shrani kot ...

Gumbi uporabniškega vmesnika so prikazani takole: Za nadaljevanje pritisnite V redu.

User Actions are indicated using a bold font. **Alt+A**: press the **Alt**-Key on your keyboard and while holding it down press the **A**-Key as well. Right drag: press the right mouse button and while holding it down *drag* the items to the new location.

Odgovor sistema in vnos preko tipkovnice sta prav tako ponazorjena z `drugačno` pisavo.

Pomembno

Pomembne opombe so označene z ikono.

Namig

Namigi za lažje delo.

Opozorilo

Kjer morate biti previdni, kaj počnete.

Pozor

Where extreme care has to be taken. Data corruption or other nasty things may occur if these warnings are ignored.

Poglavje 1. Kako začeti

This section is aimed at people who would like to find out what TortoiseSVN is all about and give it a test drive. It explains how to install TortoiseSVN and set up a local repository, and it walks you through the most commonly used operations.

1.1. Namestitev TortoiseSVN

1.1.1. Sistemske zahteve

TortoiseSVN runs on Windows Vista or higher and is available in both 32-bit and 64-bit flavours. The installer for 64-bit Windows also includes the 32-bit extension parts. Which means you don't need to install the 32-bit version separately to get the TortoiseSVN context menu and overlays in 32-bit applications.

Support for Windows 98, Windows ME and Windows NT4 was dropped in version 1.2.0, and Windows 2000 and XP up to SP2 support was dropped in 1.7.0. Support for Windows XP with SP3 was dropped in 1.9.0. You can still download and install older versions if you need them.

1.1.2. Namestitev

TortoiseSVN comes with an easy to use installer. Double click on the installer file and follow the instructions. The installer will take care of the rest. Don't forget to reboot after installation.

Pomembno

You need Administrator privileges to install TortoiseSVN. The installer will ask you for Administrator credentials if necessary.

Language packs are available which translate the TortoiseSVN user interface into many different languages. Please check Dodatek G, *Language Packs and Spell Checkers* for more information on how to install these.

If you encounter any problems during or after installing TortoiseSVN please refer to our online FAQ at https://tortoisesvn.net/faq.html.

1.2. Osnovni principi

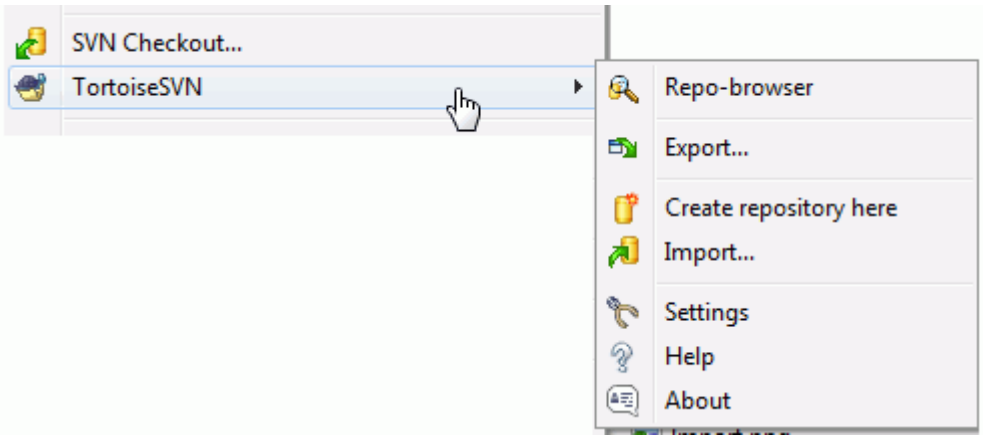
Before we get stuck into working with some real files, it is important to get an overview of how Subversion works and the terms that are used.

Skladišče
: Subversion uses a central database which contains all your version-controlled files with their complete history. This database is referred to as the *repository*. The repository normally lives on a file server running the Subversion server program, which supplies content to Subversion clients (like TortoiseSVN) on request. If you only back up one thing, back up your repository as it is the definitive master copy of all your data.

Delovna kopija
: This is where you do the real work. Every developer has his own working copy, sometimes known as a sandbox, on his local PC. You can pull down the latest version from the repository, work on it locally without affecting anyone else, then when you are happy with the changes you made commit them back to the repository.

  A Subversion working copy does not contain the history of the project, but it does keep a copy of the files as they exist in the repository before you started making changes. This means that it is easy to check exactly what changes you have made.

You also need to know where to find TortoiseSVN because there is not much to see from the Start Menu. This is because TortoiseSVN is a Shell extension, so first of all, start Windows Explorer. Right click on a folder in Explorer and you should see some new entries in the context menu like this:



Slika 1.1. Menu TortoiseSVN za datoteke brez različic

1.3. Go for a Test Drive

This section shows you how to try out some of the most commonly used features on a small test repository. Naturally it doesn't explain everything - this is just the Quick Start Guide after all. Once you are up and running you should take the time to read the rest of this user guide, which takes you through things in much more detail. It also explains more about setting up a proper Subversion server.

1.3.1. Creating a Repository

For a real project you will have a repository set up somewhere safe and a Subversion server to control it. For the purposes of this tutorial we are going to use Subversion's local repository feature which allows direct access to a repository created on your hard drive without needing a server at all.

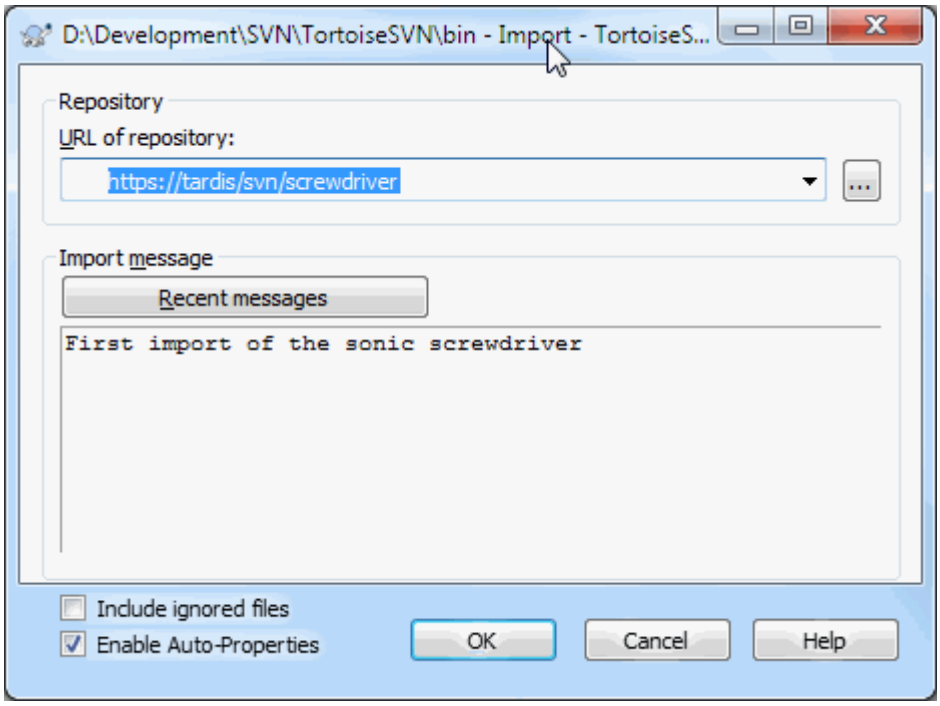
First create a new empty directory on your PC. It can go anywhere, but in this tutorial we are going to call it `C:\svn_repos`. Now right click on the new folder and from the context menu choose TortoiseSVN → Create Repository here.... The repository is then created inside the folder, ready for you to use. We will also create the default internal folder structure by clicking the Create folder structure button.

Pomembno

The local repository feature is very useful for test and evaluation but unless you are working as a sole developer on one PC you should always use a proper Subversion server. It is tempting in a small company to avoid the work of setting up a server and just access your repository on a network share. Don't ever do that. You will lose data. Read Razdelek 3.1.4, “Dostop do skladišča na deljenem omrežnem pogonu” to find out why this is a bad idea, and how to set up a server.

1.3.2. Importing a Project

Now we have a repository, but it is completely empty at the moment. Let's assume I have a set of files in `C:\Projects\Widget1` that I would like to add. Navigate to the `Widget1` folder in Explorer and right click on it. Now select TortoiseSVN → Import... which brings up a dialog



Slika 1.2. Okno za uvažanje

A Subversion repository is referred to by URL, which allows us to specify a repository anywhere on the Internet. In this case we need to point to our own local repository which has a URL of `file:///c:/svn_repos/trunk`, and to which we add our own project name `Widget1`. Note that there are 3 slashes after `file:` and that forward slashes are used throughout.

The other important feature of this dialog is the Import Message box which allows you to enter a message describing what you are doing. When you come to look through your project history, these commit messages are a valuable guide to what changes have been made and why. In this case we can say something simple like “Import the Widget1 project”. Click on OK and the folder is added to your repository.

1.3.3. Checking out a Working Copy

Now that we have a project in our repository, we need to create a working copy to use for day-to-day work. Note that the act of importing a folder does not automatically turn that folder into a working copy. The Subversion term for creating a fresh working copy is `Checkout`. We are going to checkout the Widget1 folder of our repository into a development folder on the PC called `C:\Projects\Widget1-Dev`. Create that folder, then right click on it and select TortoiseSVN → Checkout.... Then enter the URL to checkout, in this case `file:///c:/svn_repos/trunk/Widget1` and click on OK. Our development folder is then populated with files from the repository.

Pomembno

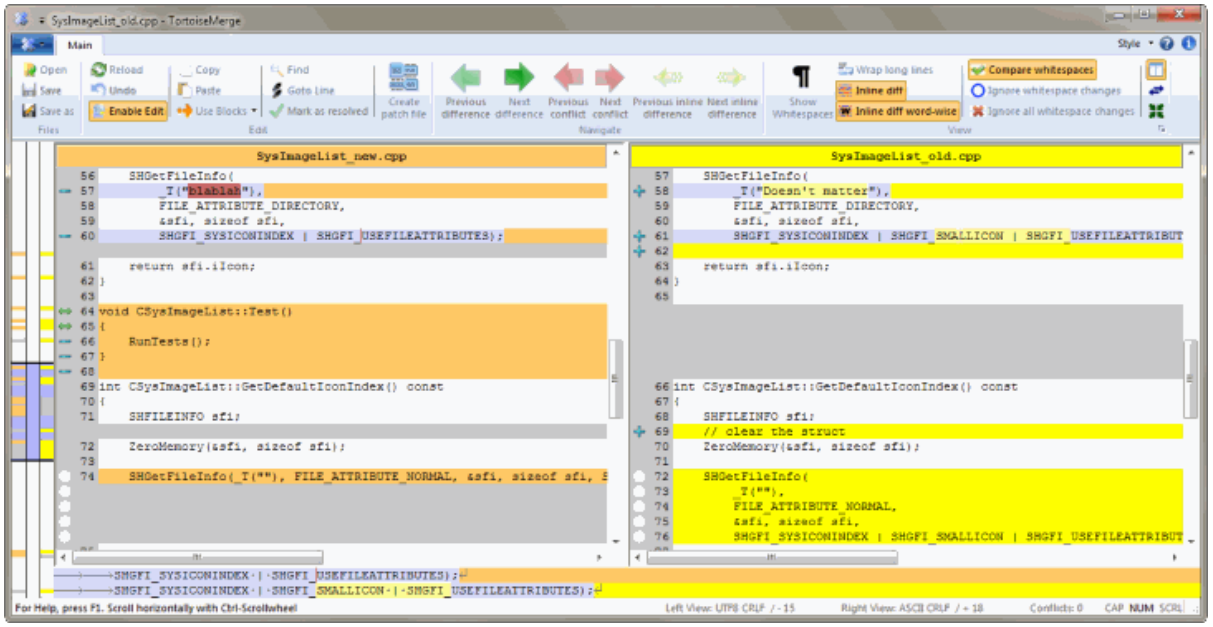
In the default setting, the checkout menu item is not located in the TortoiseSVN submenu but is shown at the top explorer menu. TortoiseSVN commands that are not in the submenu have `SVN` prepended: SVN Checkout...

You will notice that the appearance of this folder is different from our original folder. Every file has a green check mark in the bottom left corner. These are TortoiseSVN's status icons which are only present in a working copy. The green state indicates that the file is unchanged from the version in the repository.

1.3.4. Making Changes

Time to get to work. In the `Widget1-Dev` folder we start editing files - let's say we make changes to `Widget1.c` and `ReadMe.txt`. Notice that the icon overlays on these files have now changed to red, indicating that changes have been made locally.

But what are the changes? Right click on one of the changed files and select TortoiseSVN → Diff. TortoiseSVN's file compare tool starts, showing you exactly which lines have changed.



Slika 1.3. File Difference Viewer

OK, so we are happy with the changes, let's update the repository. This action is referred to as a `Commit` of the changes. Right click on the `Widget1-Dev` folder and select TortoiseSVN → Commit. The commit dialog lists the changed files, each with a checkbox. You might want to choose only a subset of those files, but in this case we are going to commit the changes to both files. Enter up a message to describe what the change is all about and click on OK. The progress dialog shows the files being uploaded to the repository and you're done.

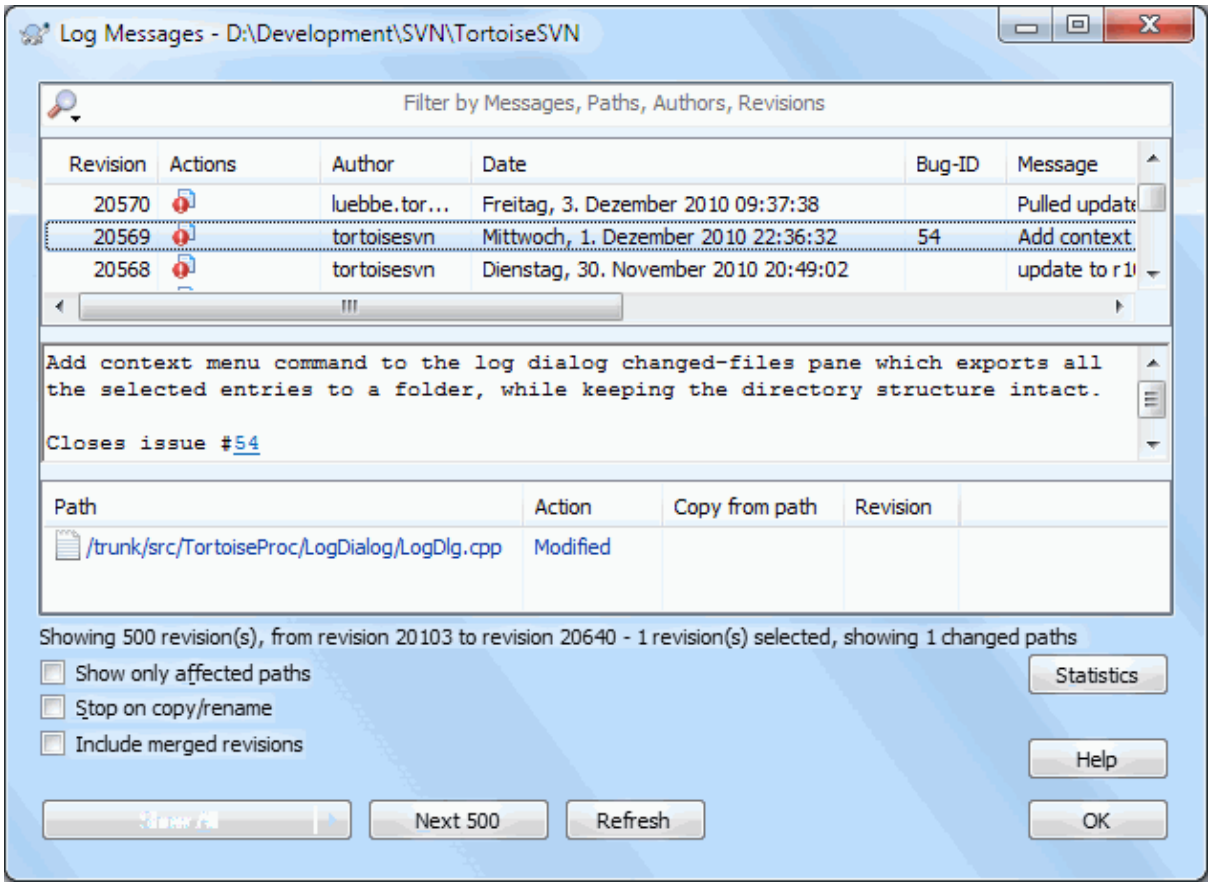
1.3.5. Adding More Files

As the project develops you will need to add new files - let's say you add some new features in `Extras.c` and add a reference in the existing `Makefile`. Right click on the folder and TortoiseSVN → Add. The Add dialog now shows you all unversioned files and you can select which ones you want to add. Another way of adding files would be to right click on the file itself and select TortoiseSVN → Add.

Now when you go to commit the folder, the new file shows up as *Added* and the existing file as *Modified*. Note that you can double click on the modified file to check exactly what changes were made.

1.3.6. Viewing the Project History

One of the most useful features of TortoiseSVN is the Log dialog. This shows you a list of all the commits you made to a file or folder, and shows those detailed commit messages that you entered (you *did* enter a commit message as suggested? If not, now you see why this is important).



Slika 1.4. The Log Dialog

OK, so I cheated a little here and used a screenshot from the TortoiseSVN repository.

The top pane shows a list of revisions committed along with the start of the commit message. If you select one of these revisions, the middle pane will show the full log message for that revision and the bottom pane will show a list of changed files and folders.

Each of these panes has a context menu which provides you with lots more ways of using the information. In the bottom pane you can double click on a file to see exactly what changes were made in that revision. Read Razdelek 4.10, “Pogovorno okno Dnevnik” to get the full story.

1.3.7. Undoing Changes

One feature of all revision control systems is that they let you undo changes that you made previously. As you would expect, TortoiseSVN makes this easy to access.

If you want to get rid of changes that you have not yet committed and reset your file to the way it was before you started editing, TortoiseSVN → Revert is your friend. This discards your changes (to the Recycle bin, just in case) and reverts to the committed version you started with. If you want to get rid of just some of the changes, you can use TortoiseMerge to view the differences and selectively revert changed lines.

If you want to undo the effects of a particular revision, start with the Log dialog and find the offending revision. Select Context Menu → Revert changes from this revision and those changes will be undone.

1.4. Moving On ...

This guide has given you a very quick tour of some of TortoiseSVN's most important and useful features, but of course there is far more that we haven't covered. We strongly recommend that you take the time to read the rest of this manual, especially Poglavje 4, *Dnevna uporaba* which gives you a lot more detail on day-to-day operations.

We have taken a lot of trouble to make sure that it is both informative and easy to read, but we recognise that there is a lot of it! Take your time and don't be afraid to try things out on a test repository as you go along. The best way to learn is by using it.

Poglavje 2. Basic Version-Control Concepts

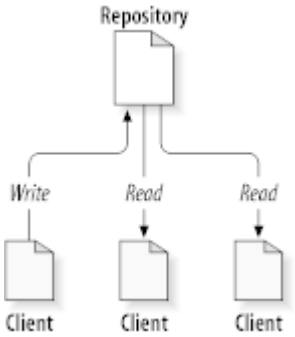
To poglavje je rahlo spremenjena verzija istega poglavja v knjigi The Subversion book. Spletna različica knjige je na voljo na tem naslovu: http://svnbook.red-bean.com/.

To poglavje je kratek uvod v Subversion. Če se z nadzorom različic še niste srečali, potem je to poglavje za vas. Začnemo s splošnimi principi nadzora različic, pregledamo posebnosti sistema Subversion in na enostavnih primerih pogledamo, kako se Subversion uporablja.

Čeprav primeri v tem poglavju prikazujejo, kako si uporabniki delijo programsko izvorno kodo, upoštevajte, da lahko Subversion upravlja katerekoli datoteke - ni omejen samo na pomoč računalniškim programerjem.

2.1. Skladišče

Subversion je centraliziran sistem za souporabo informacij. Njegovo bistvo je *skladišče*, centralno mesto za shranjevanje podatkov. Skladišče shranjuje informacije v obliki *drevesne strukture datotečnega sistema*, kar je tipična hierarhija datotek in map. S skladiščem se lahko poveže poljubno število *odjemalcev* in bere ali zapisuje datoteke. S pisanjem podatkov da odjemalec te podatke na voljo drugim; z branjem podatkov odjemalec dobi infomacije od drugih.



Slika 2.1. Tipičen sistem odjemalec/strežnik

Zakaj je to zanimivo? Po dosedanji razlagi izgleda to kot tipičen datotečni strežnik. In v resnici skladišče *je* neke vrste datotečni strežnik. Subversion je poseben zaradi dejstva, da si *zapomni vsako spremembo*, ki je bila kadarkoli zapisana: vsako spremembo vsake datoteke, pa tudi spremembe strukture map, n. pr. dodajanje, brisanje in preureditev datotek in map.

When a client reads data from the repository, it normally sees only the latest version of the filesystem tree. But the client also has the ability to view *previous* states of the filesystem. For example, a client can ask historical questions like, “ what did this directory contain last Wednesday? ”, or “ who was the last person to change this file, and what changes did they make? ” These are the sorts of questions that are at the heart of any *version control system*: systems that are designed to record and track changes to data over time.

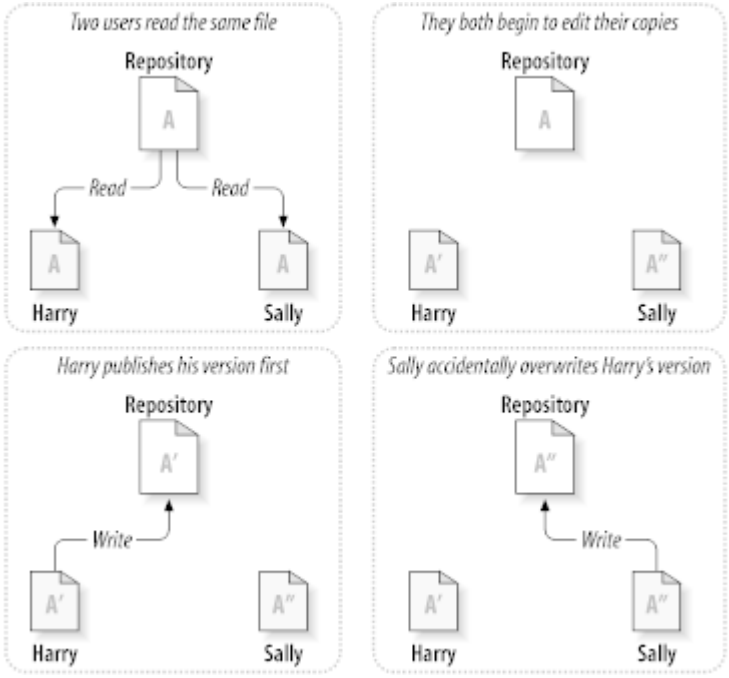
2.2. Modeli nadzora različic

Vsi sistemi nadzora različic morajo rešiti isto osnovno težavo: kako bo sistem omogočal souporabo informacij, hkrati pa preprečil, da bi si uporabniki skakali v zelje. Vse prelahko je povoziti spremembe ostalih uporabnikov v skladišču.

2.2.1. Težave souporabe datotek

Zamislite si naslednji scenarij: imamo dva sodelavca, Harryja in Sally. Odločita se, da bosta spremenjala isto datoteko iz skladišča. Če spremembe najprej shrani Harry, potem je možno, da Sally (čez nekaj trenutkov)

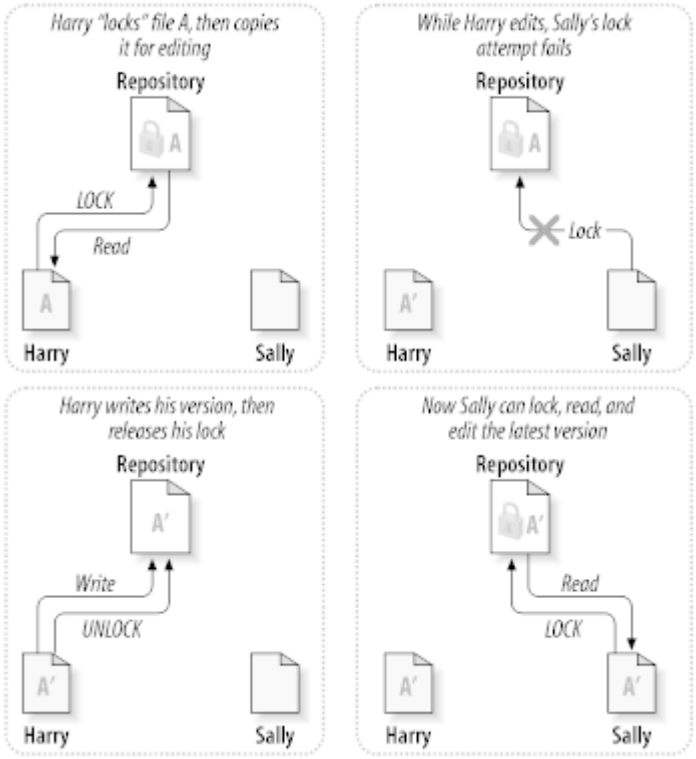
pomotoma prepiše te spremembe s svojo novo različico datoteke. Harryjeva datoteka sicer ne bo izgubljena, saj si sistem zapomni vsako spremembo, *ne bodo* pa te spremembe prisotne v Sallyjini datoteki, saj sploh ne ve zanje. Harryjevo delo je tako kljub vsemu izgubljeno - oziroma vsaj ni na voljo v zadnji različici - in to po vsej verjetnosti po pomoti. Takšnim situacijam se želimo izogniti.



Slika 2.2. Težava, ki se ji je potrebno izogniti

2.2.2. Rešitev zakleni-spremeni-odkleni

Veliko sistemov za nadzor različic uporablja model *zakleni-spremeni-odkleni*, kar predstavlja zelo enostavno rešitev problema. V takšnem sistemu lahko datoteko spreminja samo en uporabnik naenkrat. Pred spreminjanjem datoteke jo mora Harry *zakleniti*. Zaklepanje datoteke je podobno izposoji knjige v knjižnici; če Harry zaklene datoteko, potem Sally ne more narediti nobenih sprememb na tej datoteki. Če poskuša datoteko zakleniti, ji bo skladišče to onemogočilo. Datoteko lahko le bere in čaka, da jo bo Harry nehal urejati in sprostil zaklep. Potem ko Harry odklene datoteko, jo lahko zaklene Sally in jo začne urejati.



Slika 2.3. Rešitev zakleni-spremeni-odkleni

Model zakleni-spremeni-odkleni uporabnike omejuje in jih ovira pri delu:

- *Zaklepanje lahko povzroča skrbniške težave.* Včasih Harry zaklene datoteko in nanjo pozabi. Medtem je Sally, ki želi spremeniti isto datoteko, prisiljena čakati. Harry gre na počitnice, Sally pa mora po pomoč k skrbniku sistema, da ji odklene datoteko. Takšna situacija povzroči precej nepotrebnih zamud in izgube časa.

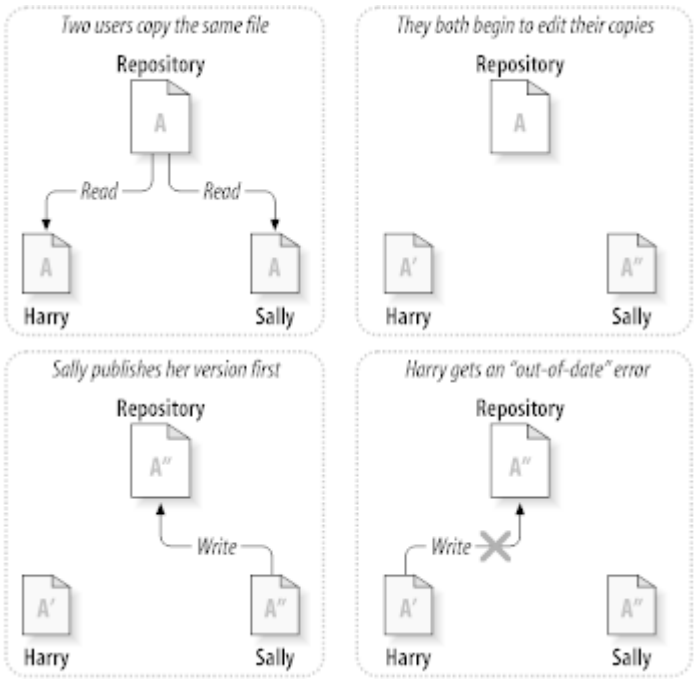
- *Zaklepanje lahko povzroči nepotrebno delo eden za drugim.* Harry popravlja začetni del datoteke, Sally pa želi popravljati zadnji del iste datoteke. Te spremembe se ne prekrivajo. Oba uporabnika bi lahko datoteko spreminjala sočasno brez posledic, pod pogojem, da bi spremembe pravilno spojila skupaj. Nobene potrebe ni, da v takšni situaciji delata eden za drugim.

- *Zaklepanje ustvarja lažen občutek varnosti.* Predstavljajte si, da Harry zaklene in spreminja datoteko A, Sally pa hkrati zaklene in spreminja datoteko B. Recimo, da sta datoteki A in B odvisni ena od druge, in spremembe, narejene na posamezni datoteki, so semantično nezdružljive. Kar naenkrat datoteki A in B skupaj ne funkcionirata pravilno. Sistem zaklepanja je nemočen pri preprečevanju težave - hkrati pa ustvarja lažen občutek varnosti. Harry in Sally si mislita, da z zaklepanjem datotek začenjata varno, izolirano operacijo, kar povzroči, da se sploh ne pogovarjata o možni nezdružljivosti datotek.

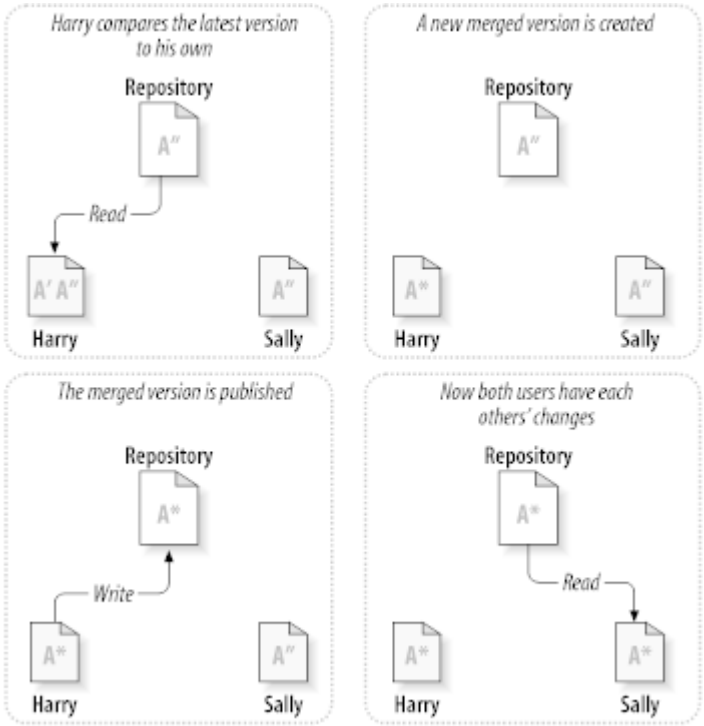
2.2.3. Rešitev kopiraj-spremeni-spoji

Subversion, CVS, and other version control systems use a *copy-modify-merge* model as an alternative to locking. In this model, each user's client reads the repository and creates a personal *working copy* of the file or project. Users then work in parallel, modifying their private copies. Finally, the private copies are merged together into a new, final version. The version control system often assists with the merging, but ultimately a human being is responsible for making it happen correctly.

Poglejmo primer. Harry in Sally ustvarita delovni kopiji istega projekta, prenesenega iz skladišča. Delata istočasno in naredita spremembe v isti datoteki `A` v svojih delovnih kopijah. Sally prva shrani spremembe v skladišče. Ko Harry kasneje poskusi shraniti svoje spremembe, ga skladišče obvesti, da je njegova datoteka A *zastarela*. Z drugimi besedami: datoteka A se je od takrat, ko jo je Harry prevzel iz skladišča, spremenila. Zato Harry zahteva od svojega odjemalca, da nove spremembe v skladišču *spoji* z datoteko A v njegovi delovni kopiji. Mogoče se spremembe ne prepletajo. Ko ima obe verziji združeni, shrani datoteko v skladišče.



Slika 2.4. Rešitev kopiraj-spremeni-spoji



Slika 2.5. ...Kopiraj-spremeni-spoji (nadaljevanje)

Kaj pa, če se Sallyjine spremembe *vseeno* prekrivajo s Harryjevimi? Nastane *sporna* situacija, ki pa običajno ne povzroča problemov. Ko Harry zahteva od svojega odjemalca, naj spoji zadnje spremembe v delovno kopijo, je njegova datoteka A označena kot sporna: Harry bo lahko videl oba nabora spornih sprememb in imel možnost izbrati pravilne. Upoštevajte, da program ne more samodejno reševati sporov; samo ljudje smo sposobni razmišljati in narediti ustrezne inteligentne odločitve. Ko Harry ročno reši prekrivajoče se spremembe (mogoče se je ob tem moral celo posvetovati s Sally!), lahko varno shrani spojeno datoteko v skladišče.

Model kopiraj-spremeni-spoji se mogoče zdi nekoliko kompliciran, vendar se v praksi izkaže, da funkcionira zelo dobro. Uporabniki lahko delajo sočasno, nikoli jim ni potrebno čakati nekoga drugega. Izkaže se, da se v primerih, ko delajo na istih datotekah, večina sprememb ne prekriva. Spori so redki. In čas, ki ga uporabniki porabijo za reševanje sporov, je mnogo krajši, kot bi bila izguba časa zaradi sistema zaklepanja.

Na koncu pridemo do enega samega ključnega dejavnika: komunikacija med uporabniki. Kjer se uporabniki malo pogovarjajo, se poveča število sintaktičnih in semantičnih sporov. Noben sistem ne more prisiliti uporabnikov, da bi idealno komunicirali med sabo in noben sistem ne more zaznati semantičnih sporov. Torej nima nobenega smisla živeti v lažnem prepričanju, da bo sistem zaklepanja preprečil spore. V praksi se izkaže, da zaklepanje zmanjša produktivnost bolj kot karkoli drugega.

Obstaja pa primer, v katerem se model zakleni-spremeni-odkleni izkaže za boljšega in sicer pri datotekah, ki se jih ne da spajati. Na primer: če vaše skladišče vsebuje grafične datoteke in dva uporabnika naredita spremembe na isti datoteki naenkrat, se teh dveh sprememb ne da združiti. Eden od obeh uporabnikov bo spremembe izgubil.

2.2.4. Kaj naredi Subversion?

Subversion po privzetih nastavitvah uporablja model kopiraj-spremeni-spoji in v veliko primerih je to vse, kar potrebujete. Od različice 1.2 naprej pa Subversion podpira tudi zaklepanje datotek. Če torej delate na datotekah, ki se jih ne da spajati ali če ste takšen model prisiljeni uporabljati zaradi pravil, ki veljajo v vašem podjetju, vam Subversion ponuja možnosti, ki jih potrebujete.

2.3. Subversion v akciji

2.3.1. Delovne kopije

O delovnih kopijah ste že brali. Sedaj bomo prikazali, kako jih odjemalec Subversion ustvari in uporablja.

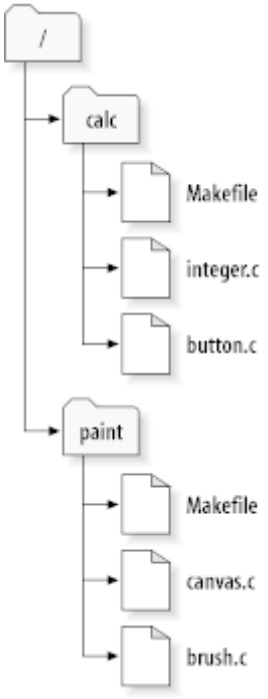
Delovna kopija sistema Subversion je običajna struktura map na vašem krajevnem sistemu, ki vsebuje datoteke. Datoteke lahko poljubno spreminjate. Če so to datoteke izvorne kode, lahko iz njih povsem običajno prevedete program. Vaša delovna kopija je vaše zasebno delovno območje: Subversion ne bo v vašo delovno kopijo nikoli vnašal sprememb, ki so jih naredili drugi uporabniki, prav tako vaših sprememb ne bo dal na voljo drugim, razen ko boste to izrecno zahtevali od njega.

Ko ste naredili določene spremembe v datotekah v delovni kopiji in jih preverili, uporabite ukaze sistema Subversion, da spremembe *objavite* (s pisanjem v skladišče), tako da so vidne ostalim sodelavcem, ki delajo na projektu. Če ostali sodelavci naredijo spremembe na datotekah, vam Subversion omogoča, da te spremembe spojite v delovno kopijo (z branjem iz skladišča).

A working copy also contains some extra files, created and maintained by Subversion, to help it carry out these commands. In particular, your working copy contains a subdirectory named `.svn`, also known as the working copy *administrative directory* . The files in this administrative directory help Subversion recognize which files contain unpublished changes, and which files are out-of-date with respect to others' work. Prior to 1.7 Subversion maintained `.svn` administrative subdirectories in every versioned directory of your working copy. Subversion 1.7 takes a completely different approach and each working copy now has only one administrative subdirectory which is an immediate child of the root of that working copy.

Tipično skladišče sistema Subversion pogosto vsebuje datoteke (ali izvorno kodo) za več projektov; običajno je vsak projekt podmapa v drevesni strukturi skladišča. Pri takšni ureditvi bo uporabnikova delovna kopija običajno ustrezala določenemu podrevesu skladišča.

Recimo, da imate skladišče z dvema projektoma programske opreme.



Slika 2.6. Datotečni sistem skladišča

Z drugimi besedami, korenska mapa skladišča ima dve podmapi: `paint` in `calc`.

To get a working copy, you must *check out* some subtree of the repository. (The term *check out* may sound like it has something to do with locking or reserving resources, but it doesn't; it simply creates a private copy of the project for you.)

Suppose you make changes to `button.c`. Since the `.svn` directory remembers the file's modification date and original contents, Subversion can tell that you've changed the file. However, Subversion does not make your changes public until you explicitly tell it to. The act of publishing your changes is more commonly known as *committing* (or *checking in*) changes to the repository.

Da bi omogočili dostop do svojih sprememb ostalim, uporabite ukaz **objavi**.

Sedaj so spremembe v datoteki `button.c` objavljene v skladišču; če nek drug uporabnik prevzame delovno kopijo iz mape `/calc`, bo te spremembe videl v zadnji različici datoteke.

Predpostavimo, da imate sodelavko Sally, ki je prevzela delovno kopijo `/calc` sočasno z vami. Ko objavite spremembe v datoteki `button.c`, se Sallyjina delovna kopija ne spremeni; Subversion spreminja delovne kopije le na zahtevo uporabnika.

Za posodobitev projekta Sally od sistema Subversion zahteva, da *posodobi* njeno delovno kopijo z uporabo ukaza **update**. Ta spoji vaše spremembe v njeno delovno kopijo, prav tako pa tudi vse ostale spremembe, ki so bile objavljene po njenem prevzemu.

Upoštevajte, da Sally ni potrebno povedati, katere datoteke želi posodobiti; Subversion uporabi informacije v mapi `.svn` in dodatne informacije iz skladišča, da se odloči, katere datoteke je potrebno posodobiti.

2.3.2. Naslovi URL skladišča

Do skladišč Subversion lahko dostopate na več različnih načinov - na krajevnem disku ali preko različnih omrežnih protokolov. Lokacija skladišča pa je vedno navedena z naslovom URL. Shema naslova URL pove način dostopa:

Shema	Način dostopa
`file://`	Neposreden dostop do skladišča na krajevnem ali omrežnem disku.

Shema	Način dostopa
`http://`	Dostop do strežnika Apache s sistemom Subversion preko protokola WebDAV.
`https://`	Enako kot `http://`, vendar z enkripcijo SSL.
`svn://`	Dostop TCP/IP brez avtentikacije preko protokola po meri do strežnika `svnserve`
`svn+ssh://`	avtenticiran, enkriptiran dostop TCP/IP preko protokola po meri do strežnika `svnserve`

Tabela 2.1. Naslovi URL za dostop do skladišča

V večini primerov se v sistemu Subversion za naslove URL uporablja standardna sintaksa, kar omogoča, da se ime strežnika in številka vrat pojavljata kot del naslova URL. Metoda dostopa `file://` se običajno uporablja za lokalni dostop, možno pa jo je uporabljati tudi za poti UNC. URL je naslednje oblike: `file://gostitelj/pat/do/skladišča`. Za krajevne dostope dela `gostitelj` ne navedemo ali pa navedemo `localhost`. Iz tega razloga se krajevne poti običajno zapisujejo s tremi poševnicami: `file:///pat/do/skladišča`.

Poleg tega morajo uporabniki sheme `file://` na operacijskem sistemu Windows uporabljati neuradno "standardno" sintakso za dostopanje do skladišč, ki so na istem računalniku, vendar na drugem pogonu, kot je trenutni delovni pogon odjemalca. Naslednji dve različici sintakse poti URL sta pravilni; `X` je pogon, na katerem se nahaja skladišče:

```
file:///X:/path/to/repos
...
file:///X|/path/to/repos
...
```

Upoštevajte, da se za naslove URL uporablja poševnica, čeprav je znak za ločitev posameznih delov poti v sistemu Windows obrnjena poševnica.

You can access a FSFS repository via a network share, but this is *not* recommended for various reasons:

- You are giving direct write access to all users, so they could accidentally delete or corrupt the repository file system.
- Not all network file sharing protocols support the locking that Subversion requires. One day you will find your repository has been subtly *corrupted*.
- You have to set the access permissions in just the right way. SAMBA is particularly difficult in this respect.
- If one person installs a newer version of the client which upgrades the repository format, then everyone else will be unable to access the repository until they also upgrade to the new client version.

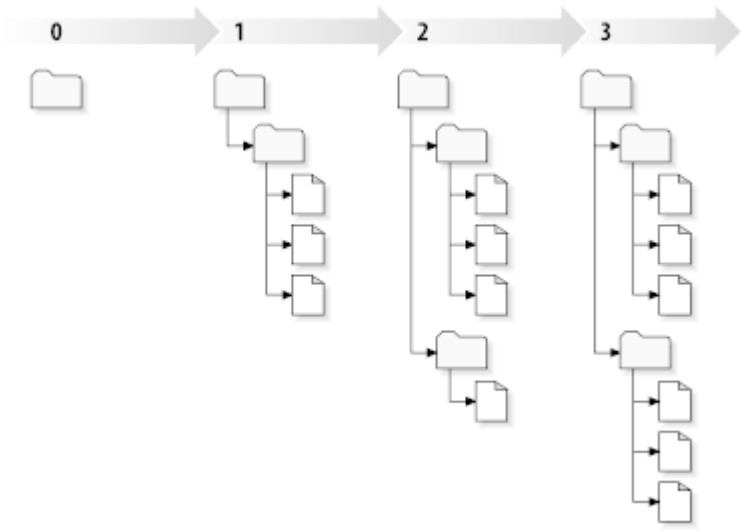
2.3.3. Revizije

Operacija **svn commit** objavi spremembe poljubnega števila datotek in map v eni atomični transakciji. V delovni kopiji lahko spreminjate vsebine datotek, ustvarjate, brišete, preimenujete in kopirate datoteke in potem objavite celoten nabor sprememb kot celoto.

V skladišču se vsaka objava opravi kot atomična transakcija: upoštevane so vse spremembe v objavi ali pa nobena. Subversion poskuša ohraniti atomičnost zaradi sesutja aplikacij, sistema, težav z omrežjem in ostalih uporabnikov posredovanj.

Vsakokrat, ko skladišče sprejme objavo, se ustvari novo stanje drevesa dototečne strukture, imenovano *revizija*. Vsaki reviziji se predpiše unikatno naravno število, za eno večje od številke prejšnje revizije. Začetna revizija novo ustvarjenega skladišča ima številko 0 in vsebuje zgolj korensko mapo.

Dober način za predstavitev skladišča je vrsta dreves. Predstavljajte si polje številk revizij, ki se začne z 0 in se razteza od leve proti desni. Iz vsake številke revizije visi drevo datotečnega sistema in vsako drevo je “posnetek” izgleda skladišča po vsaki objavi.



Slika 2.7. Skladišče

> **Globalne številke revizij**
>
> Unlike those of many other version control systems, Subversion's revision numbers apply to *entire trees* , not individual files. Each revision number selects an entire tree, a particular state of the repository after some committed change. Another way to think about it is that revision N represents the state of the repository filesystem after the Nth commit. When a Subversion user talks about ``revision 5 of `foo.c`", they really mean ``foo.c as it appears in revision 5." Notice that in general, revisions N and M of a file do *not* necessarily differ!

Pomembno je poudariti, da delovne kopije niso vedno odraz samo eno revizije v skladišču; vsebujejo lahko datoteke več različnih revizij. Primer: recimo, da prevzamete delovno kopijo iz skladišča, kjer zadnja shranjena revizija nosi številko 4:

```
calc/Makefile:4
integer.c:4
button.c:4
```

V tem trenutku delovna kopija ustreza točno reviziji številka 4 v skladišču. Denimo, da naredite spremembe v datoteki `button.c` in objavite spremembo. Denimo, da se med tem niso zgodile nobene druge objave. Vaša objava bo ustvarila revizijo 5 v skladišču in vaša delovna kopija bo izgledala takole:

```
calc/Makefile:4
integer.c:4
button.c:5
```

Denimo, da v tem trenutku Sally objavi spremembo datoteke `integer.c` in s tem ustvari revizijo 6. Če uporabite ukaz **svn update**, ki posodobi vašo delovno kopijo, bo le-ta izgledala takole:

```
calc/Makefile:6
```

```
integer.c:6
button.c:6
```

Sallyjine spremembe datoteke `integer.c` se bodo pojavile v vaši delovni kopiji in v datoteki `button.c` bodo še vedno vaše spremembe. V tem primeru je besedilo datoteke `Makefile` identično v revizijah 4, 5 in 6, vendar bo Subversion označil vašo delovno kopijo datoteke `Makefile` s številko revizije 6 in s tem povedal, je datoteka trenutna. Torej: ko boste naredili posodobitev vrhnje datoteke delovne kopije, bo le-ta ustrezala točno eni reviziji v skladišču.

2.3.4. Kako delovne kopije spremljajo skladišče

Za vsako datoteko v delovni kopiji Subversion shrani dve pomembni informaciji v skrbniško področje `.svn/`:

- what revision your working file is based on (this is called the file's *working revision*), and
- časovni žig, ki pove, kdaj je bila krajevna kopija nazadnje posodobljena iz skladišča.

S temi informacijami in komunikacijo s skladiščem lahko Subversion ugotovi, v katerem od štirih stanj se delovna datoteka nahaja:

Nespremenjeno in trenutno
: Datoteka je nespremenjena v delovni kopiji, prav tako pa od prevzema datoteke ni bila objavljene nobena sprememba v skladišču. Ukaz **objavi** ne bo storil ničesar, prav tako tudi ne ukaz **posodobi**.

Krajevno spremenjeno in trenutno
: Datoteka v delovni kopiji je spremenjena, v skladišču pa ni bila objavljena nobena sprememba od osnovne revizije. Obstajajo krajevne spremembe, ki niso bile objavljene v skladišču, tako da bo ukaz **objavi** na datoteki objavil vaše spremembe, ukaz **posodobi** pa ne bo naredil ničesar.

Nespremenjeno in zastarelo
: Datoteka ni bila spremenjena v delovni kopiji, se je pa spremenila v skladišču. Da bo datoteka enaka javni reviziji, jo je potrebno posodobiti. Ukaz **objavi** ne bo naredil ničesar, ukaz **posodobi** pa bo prenesel najnovejše spremembe iz skladišča v delovno kopijo.

Krajevno spremenjeno in zastarelo
: Datoteka je bila spremenjena v delovni kopiji in v skladišču. Ukaz **objavi** ne bo uspel; prikazalo se bo sporočilo, da je datoteka *zastarela*. Datoteko je potrebno najprej posodobiti; ukaz **posodobi** bo poskušal javne spremembe spojiti s krajevnimi. Če Subversion ne more spojiti različic samodejno, reševanje spora prepusti uporabniku.

2.4. Povzetek

V tem poglavju smo pregledali nekaj osnovnih principov sistema Subversion:

- Predstavili smo koncept centralnega skladišča, delovne kopije na strani odjemalca in polja dreves revizij v skladišču.
- Videli smo nekaj primerov, kako lahko dva sodelavca uporabljata Subversion in model 'kopiraj-spremeni-spoji' za objavljanje sprememb in sprejemanje sprememb drug od drugega.
- Govorili smo o tem, kako Subversion sledi in upravlja informacije v delovni kopiji.

Poglavje 3. Skladišče

Ne glede na uporabljen protokol za dostop do skladišča morate vedno ustvariti vsaj eno skladišče. To lahko storite preko ukazne vrstice (Subversion) ali preko grafičnega vmesnika (TortoiseSVN).

Če še niste ustvarili skladišča Subversion, je sedaj čas, da to storite.

3.1. Ustvarjanje skladišča

3.1.1. Ustvarjanje skladišča z odjemalcem za ukazno vrstico

1. Ustvarite prazno mapo z imenom SVN (n.pr. `D:\SVN\`), ki se uporablja kot korenska mapa za vsa vaša skladišča.

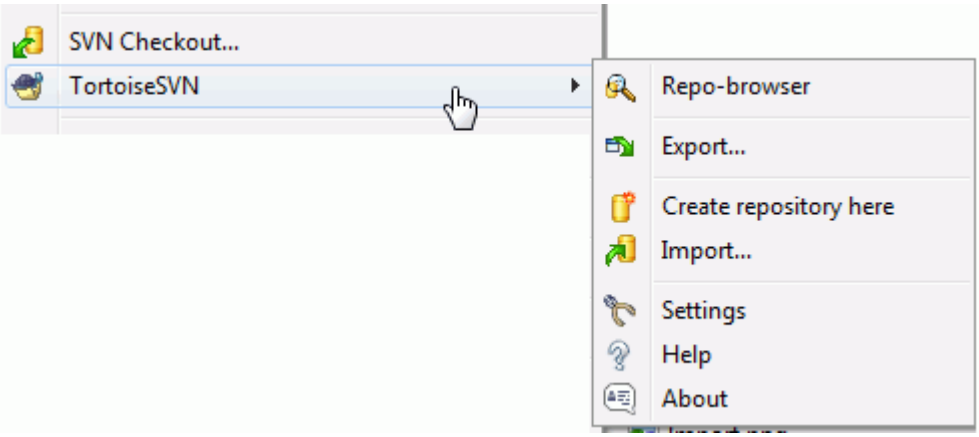
2. Ustvarite še eno mapo iz imenom `MojeNovoSkladišče` znotraj `D:\SVN\`.

3. Open the command prompt (or DOS-Box), change into `D:\SVN\` and type

```
svnadmin create --fs-type fsfs MyNewRepository
```

Sedaj imate novo skladišče na lokaciji `D:\SVN\MojeNovoSkladišče`.

3.1.2. Ustvarjanje skladišča s programom TortoiseSVN



Slika 3.1. Menu TortoiseSVN za datoteke brez različic

1. Odprite Raziskovalca

2. Ustvarite novo mapo in jo poimenujte npr. `SkladiščeSVN`

3. Right click on the newly created folder and select TortoiseSVN → Create Repository here....

   Skladišče se ustvari znotraj nove mape. *Datotek v v novi mapi ne urejajte sami!!!*. Če se pojavijo kakšne napake, preverite, ali je mapa res prazna in da ni zaščitena proti pisanju.

   You will also be asked whether you want to create a directory structure within the repository. Find out about layout options in Razdelek 3.1.5, “Postavitev skladišča”.

   TortoiseSVN will set a custom folder icon when it creates a repository so you can identify local repositories more easily. If you create a repository using the official command line client this folder icon is not assigned.

Namig

We also recommend that you don't use `file://` access at all, apart from local testing purposes. Using a server is more secure and more reliable for all but single-developer use.

3.1.3. Krajevni dostop do skladišča

Za dostop do krajevnega skladišča potrebujete le pot do skladišča. Zapomnite si, da Subversion pričakuje pot do skladišč oblike `file:///C:/SkladiščeSVN/`. Upoštevajte, da morate uporabljati poševnico in ne obrnjeno poševnico.

Za dostop do skladišča na mapi v skupni rabi v omrežju lahko uporabite preslikavo pogona ali pa pot UNC. Za poti UNC je oblika naslova URL `file://ImeStrežnika/pot/do/skladišča/`. Upoštevajte, da je potrebno na začetku uporabiti le dve poševnici.

Pred različico Subversion 1.2 so morale biti poti UNC podane v nekoliko bolj obskurni obliki `file:///\ImeStrežnika/pot/do/skladišča`. Ta oblika je še vedno možna, vendar ni priporočljiva.

3.1.4. Dostop do skladišča na deljenem omrežnem pogonu

Although in theory it is possible to put a FSFS repository on a network share and have multiple users access it using `file://` protocol, this is most definitely *not* recommended. In fact we would *strongly* discourage it, and do not support such use for various reasons:

- Prva težava je, da uporabniku omogočite popoln dostop do skladišča, tako da ga lahko ta po pomoti izbriše ali pa ga pokvari s kakšno drugo operacijo.
- Druga težava je, da vsi omrežni protokoli ne omogočajo zaklepanja datotek, ki ga Subversion za svoje delovanje potrebuje. Zato se skladišče lahko pokvari. Mogoče se to ne bo zgodilo takoj, vendar bosta slej ko prej dva uporabnika dostopala do skladišča istočasno.
- Tretja težava je, da je potrebno nastaviti ustrezne pravice na datotekah. Na navadnem omrežnem pogonu sistema Windows bo to morda celo delovalo, pri uporabi strežnika SAMBA pa nastanejo težave.
- If one person installs a newer version of the client which upgrades the repository format, then everyone else will be unable to access the repository until they also upgrade to the new client version.

Protokol `file://` je namenjen krajevnemu dostopu za enega uporabnika, predvsem za testiranje in razhroščevanje. Če želite deliti skladišče, morate *zares* razmisliti o postavitvi strežnika. To pa sploh ni tako težko, kot se sliši. Za več informacij o izbiri in navodila za postavitev preberite Razdelek 3.5, “Accessing the Repository”.

3.1.5. Postavitev skladišča

Preden uvozite podatke v skladišče, premislite, kako jih boste organizirali. Če uporabite katero od priporočenih postavitev, vam bo kasneje laže.

Obstaja nekaj standardnih, priporočenim načinov, kako organizirati skladišče Subversion. Večina ljudi naredi mapo `trunk` za “glavno vejo” razvoja, mapo `branches` za stranske veje in mapo `tags`, ki vsebuje oznake. Če se v skladišču nahaja samo en projekt, uporabniki pogosto ustvarijo naslednjo strukture vrhnjih map:

```
/trunk
/branches
/tags
```

Because this layout is so commonly used, when you create a new repository using TortoiseSVN, it will also offer to create the directory structure for you.

Če skladišče vsebuje več projektov, uporabniki pogosto indeksirajo njihovo postavitev glede na veje:

```
/trunk/paint
/trunk/calc
/branches/paint
/branches/calc
/tags/paint
/tags/calc
```

... ali po projektu:

```
/paint/trunk
/paint/branches
/paint/tags
/calc/trunk
/calc/branches
/calc/tags
```

Indeksiranje po projektih je smiselno, če projekti med sabo niso povezani in se prevzemajo neodvisno drug od drugega. Za povezane projekte, kjer želite narediti prevzem z eno potezo, ali pa v primeru, kjer so projekti povezani v en distribucijski paket, pa je indeksiranje po vejah pogosto boljša odločitev. V tem primeru imate le eno glavno vejo (trunk), relacije med podprojekti pa so lažje vidne.

Če izberete izgled `/trunk /tags /branches`, to ne pomeni, da morate prekopirati celotno glavno vejo za vsako vejo ali oznako, tako da ta struktura prinaša več prilagodljivosti.

Za nepovezane projekte boste verjetno uporabili različna skladišča. Ko objavite spremembe, se spremeni številka revizije celotnega skladišča, ne številka revizije projekta. Če imate v enem skladišču dva projekta, lahko to vodi do velikih "lukenj" pri številkah revizij. Projekta Subversion in TortoiseSVN se nahajata na istem strežniku, vendar sta postavljena v različna skladišča, kar omogoča neodvisen razvoj in reši zmedo glede številk gradenj.

Seveda vam teh postavitev ni treba upoštevati. Lahko naredite kakršnokoli variacijo, karkoli najbolj ustreza vam ali vaši ekipi sodelavcev. Postavitev, ki jo izberete, pa ni nujno trajna. Skladišče lahko kadarkoli na novo organizirate. Ker so veje in oznake navadne mape, jih lahko TortoiseSVN premakne ali preimenuje po vaših željah.

Sprememba iz enega načina postavitve v drugega se naredi z nizom premikov na strani strežnika. Če vam ni všeč, kako so stvari v skladišču organizirane, enostavno premaknite mape.

Če še niste ustvarili osnovne strukture map znotraj skladišča, je čas, da to storite sedaj. Ostajata dva načina. Če želite ustvariti le strukturo `/trunk /tags /branches`, lahko za to uporabite brskalnik po skladišču, kjer v posamičnih objavah naredite zgornje mape. Če želite izdelati globljo hierarhijo, potem je enostavneje, da strukturo map naredite na disku in jo nato v eni objavi uvozite:

1. ustvarite novo prazno mapo na trdem disku
2. ustvarite željeno strukturo map vrhnjega nivoja znotraj mape - zaenkrat ne vnašajte datotek!
3. import this structure into the repository via a right click on the folder that contains this folder structure and selecting TortoiseSVN → Import... In the import dialog enter the URL to your repository and click OK. This will import your temp folder into the repository root to create the basic repository layout.

Note that the name of the folder you are importing does not appear in the repository, only its contents. For example, create the following folder structure:

```
C:\Temp\New\trunk
```

```
C:\Temp\New\branches
C:\Temp\New\tags
```

Import `C:\Temp\New` into the repository root, which will then look like this:

```
/trunk
/branches
/tags
```

3.2. Varnostna kopija skladišča

Neglede na tip skladišča, ki ga uporabljate, je izjemno pomembno, da redno ustvarjate varnostne kopije in da jih tudi preverite. Če je s strežnikom kaj narobe, boste mogoče lahko dobili zadnjo verzijo datotek, vendar je brez skladišča zgodovina vaših datotek za vedno izgubljena.

The simplest (but not recommended) way is just to copy the repository folder onto the backup medium. However, you have to be absolutely sure that no process is accessing the data. In this context, access means *any* access at all. If your repository is accessed at all during the copy, (web browser left open, WebSVN, etc.) the backup will be worthless.

The recommended method is to run

```
svnadmin hotcopy path/to/repository path/to/backup
```

to create a copy of your repository in a safe manner. Then backup the copy.

The `svnadmin` tool is installed automatically when you install the Subversion command line client. The easiest way to get this is to check the option to include the command line tools when installing TortoiseSVN, but if you prefer you can download the latest version of command line tools directly from the *Subversion* [https://subversion.apache.org/packages.html#windows] website.

3.3. Server side hook scripts

A hook script is a program triggered by some repository event, such as the creation of a new revision or the modification of an unversioned property. Each hook is handed enough information to tell what that event is, what target(s) it's operating on, and the username of the person who triggered the event. Depending on the hook's output or return status, the hook program may continue the action, stop it, or suspend it in some way. Please refer to the chapter on *Hook Scripts* [http://svnbook.red-bean.com/en/1.8/svn.reposadmin.create.html#svn.reposadmin.create.hooks] in the Subversion Book for full details about the hooks which are implemented.

Ukazne datoteke akcij izvede strežnik, ki gosti skladišče. TortoiseSVN pa vam omogoča, da nastavite ukazne datoteke akcij na strani odjemalca. Te datoteke se izvedejo krajevno ob določenih dogodkih. Za več informacij poglejte Razdelek 4.31.8, “Ukazne datoteke akcij na strani odjemalca”.

Sample hook scripts can be found in the `hooks` directory of the repository. These sample scripts are suitable for Unix/Linux servers but need to be modified if your server is Windows based. The hook can be a batch file or an executable. The sample below shows a batch file which might be used to implement a pre-revprop-change hook.

```
rem Only allow log messages to be changed.
if "%4" == "svn:log" exit 0
echo Property '%4' cannot be changed >&2
exit 1
```

Note that anything sent to stdout is discarded. If you want a message to appear in the Commit Reject dialog you must send it to stderr. In a batch file this is achieved using `>&2`.

Overriding Hooks

If a hook script rejects your commit then its decision is final. But you can build an override mechanism into the script itself using the *Magic Word* technique. If the script wants to reject the operation it first scans the log message for a special pass phrase, either a fixed phrase or perhaps the filename with a prefix. If it finds the magic word then it allows the commit to proceed. If the phrase is not found then it can block the commit with a message like “You didn't say the magic word”. :-)

3.4. Povezave za prevzem

Če želite ponuditi skladišče Subversion tudi ostalim uporabnikom, lahko povezavo vključite v svojo spletno stran. Da jo naredite bolj dostopno, lahko dodate *prevzemno povezavo* za uporabnike TortoiseSVN.

Ko namestite TortoiseSVN, ta registrira nov protokol `tsvn:`. Ko uporabnik klikne na takšno povezavo, se samodejno odpre okno za prevzem z že izpolnjenim naslovom URL skladišča.

Če želite dodati takšno povezavo na svojo HTML stran, ji dodajte naslednjo kodo:

```
<a href="tsvn:http://project.domain.org/svn/trunk">
</a>
```

Of course it would look even better if you included a suitable picture. You can use the *TortoiseSVN logo* [https://tortoisesvn.net/images/TortoiseCheckout.png] or you can provide your own image.

```
<a href="tsvn:http://project.domain.org/svn/trunk">
<img src=TortoiseCheckout.png></a>
```

You can also make the link point to a specific revision, for example

```
<a href="tsvn:http://project.domain.org/svn/trunk?100">
</a>
```

3.5. Accessing the Repository

To use TortoiseSVN (or any other Subversion client), you need a place where your repositories are located. You can either store your repositories locally and access them using the `file://` protocol or you can place them on a server and access them with the `http://` or `svn://` protocols. The two server protocols can also be encrypted. You use `https://` or `svn+ssh://`, or you can use `svn://` with SASL.

If you are using a public hosting service such as *SourceForge* [https://sourceforge.net] or your server has already been setup by someone else then there is nothing else you need to do. Move along to Poglavje 4, *Dnevna uporaba*.

If you don't have a server and you work alone, or if you are just evaluating Subversion and TortoiseSVN in isolation, then local repositories are probably your best choice. Just create a repository on your own PC as described earlier in Poglavje 3, *Skladišče*. You can skip the rest of this chapter and go directly to Poglavje 4, *Dnevna uporaba* to find out how to start using it.

If you were thinking about setting up a multi-user repository on a network share, think again. Read Razdelek 3.1.4, “Dostop do skladišča na deljenem omrežnem pogonu” to find out why we think this is a bad idea. Setting up a server is not as hard as it sounds, and will give you better reliability and probably speed too.

More detailed information on the Subversion server options, and how to choose the best architecture for your situation, can be found in the Subversion book under *Server Configuration* [http://svnbook.red-bean.com/en/1.8/svn.serverconfig.html].

In the early days of Subversion, setting up a server required a good understanding of server configuration and in previous versions of this manual we included detailed descriptions of how to set up a server. Since then things have become easier as there are now several pre-packaged server installers available which guide you through the setup and configuration process. These links are for some of the installers we know about:

- *VisualSVN* [https://www.visualsvn.com/server/]
- *CollabNet* [https://www.collab.net/products/subversion]

You can always find the latest links on the *Subversion* [https://subversion.apache.org/packages.html] website.

You can find further How To guides on the *TortoiseSVN* [https://tortoisesvn.net/usefultips.html] website.

Poglavje 4. Dnevna uporaba

Ta dokument opisuje dnevno uporabo odjemalca TortoiseSVN. To *ni* uvod v sisteme za nadzor različic, prav tako pa *ni* uvod v Subversion (SVN). To poglavje uporabite, ko veste, kaj želite narediti, vendar se ne spomnite natančno, kako se to naredi.

Če potrebujete uvod v nadzor različic s sistemom Subversion, vam priporočamo odlično knjigo: *Version Control with Subversion* [http://svnbook.red-bean.com/].

Tako kot programa TortoiseSVN in Subversion je tudi ta dokument "v delu". Če najdete kakšne napake, jih sporočite na dopisni seznam, da bomo lahko dokumentacijo dopolnili. Nekateri posnetki zaslonov v Dnevni uporabi so lahko drugačni, kot so trenutno v aplikaciji. Prosimo, oprostite. Aplikacijo TortoiseSVN razvijamo v svojem prostem času.

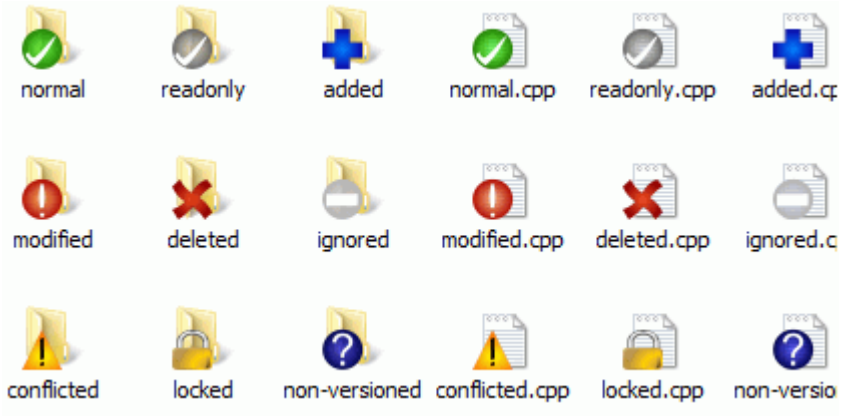
Da bi se čimveč naučili iz poglavja Dnevna uporaba:

- morate imeti TortoiseSVN že nameščen,
- morate poznati sisteme za nadzor različic,
- morate poznati osnove sistema Subversion,
- imeti morate nameščen strežnik in/ali dostop do skladišča Subversion.

4.1. General Features

This section describes some of the features of TortoiseSVN which apply to just about everything in the manual. Note that many of these features will only show up within a Subversion working copy.

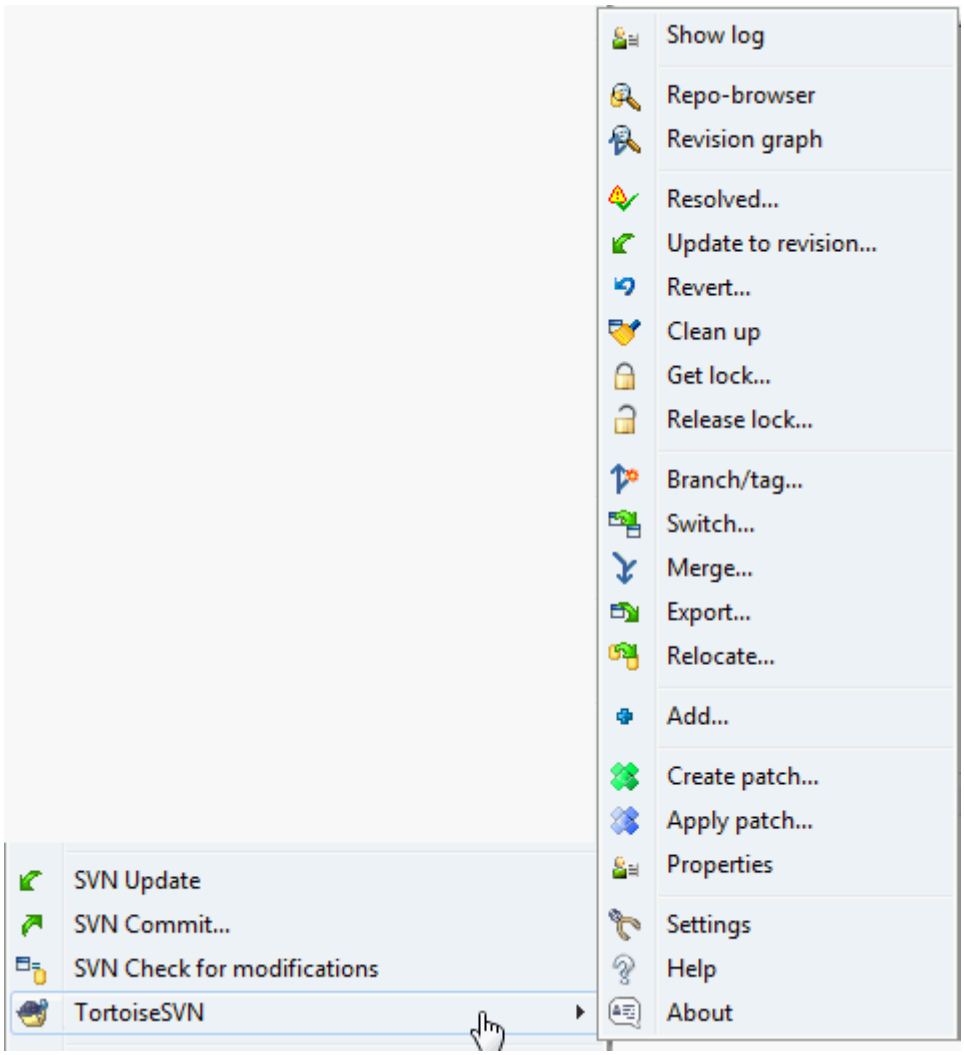
4.1.1. Prekrivne ikone



Slika 4.1. Raziskovalec prikaže prekrivne ikone

Ena izmed vizuelno najbolj opaznih zmožnosti TortoiseSVN so prekrivne ikone, ki se pojavijo na datotekah v delovni kopiji. Z njihovo pomočjo že na prvi pogled vidite, katere datoteke so spremenjene. Preberite Razdelek 4.7.1, “Prekrivne ikone” za več informacij o tem, kaj različne ikone predstavljajo.

4.1.2. Kontekstni meni



Slika 4.2. Kontekstni meni za mapo pod nadzorom različic

Vsi ukazi TortoiseSVN se kličejo iz kontekstnega menija Raziskovalca. Večina je vidna, ko desno kliknete na datoteko ali mapo. Ukazi, ki so na voljo, so odvisni od tega, ali je datoteka ali mapa oziroma njihova nadrejena mapa pod nadzorom različic ali ne. Meni TortoiseSVN lahko vidite tudi kot del glavnega menija Datoteka v Raziskovalcu.

Namig

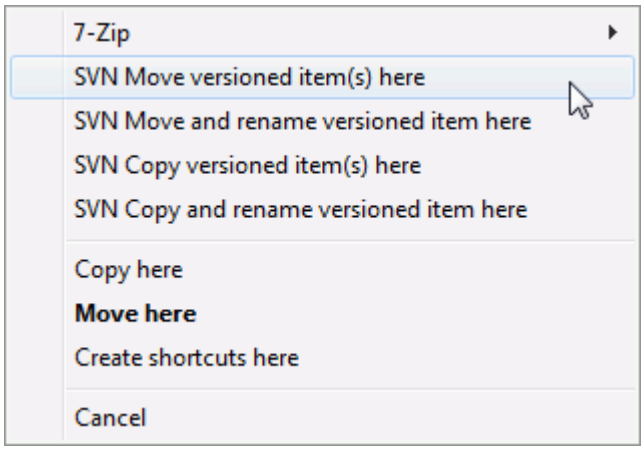
Some commands which are very rarely used are only available in the extended context menu. To bring up the extended context menu, hold down the **Shift** key when you right click.

V nekaterih primerih lahko vidite več elementov TortoiseSVN. To ni napaka!

Slika 4.3. Kontekstni meni v Raziskovalcu za bližnjico v mapi pod nadzorom različic

Ta primer opisuje bližnjico za datoteke brez različic. Znotraj mape pod nadzorom in v meniju Raziskovalca so *trije* elementi sistema TortoiseSVN. Eden je za mapo, eden za bližnjico samo in tretji za objekt, na katerega bližnjica kaže. Da bi lažje ločevali med njimi, imajo ikone v spodnjem desnem kotu indikator, ki pove, ali se element menija nanaša na datoteko, mapo, bližnico ali na več elementov.

4.1.3. Povleci in spusti



Slika 4.4. Meni ob premikanju mape, ki je pod nadzorom različic

Drugi ukazi so na voljo kot meni, ko z desnim gumbom povlečete datoteke ali mape na novo lokacijo znotraj delovne kopije ali ko z desnim gumbom povlečete datoteko ali mapo brez različic v mapo, ki je pod nadzorom.

4.1.4. Pogoste bližnjice

Nekatere pogoste operacije imajo znane bližnjice sistema Windows, vendar se ne pojavljajo na gumbih ali v menijih. Če vam ne uspe ugotoviti, kako se naredi kaj preprostega, n. pr. kako se osveži pogled, poglejte sem.

F1
: Pomoč, seveda.

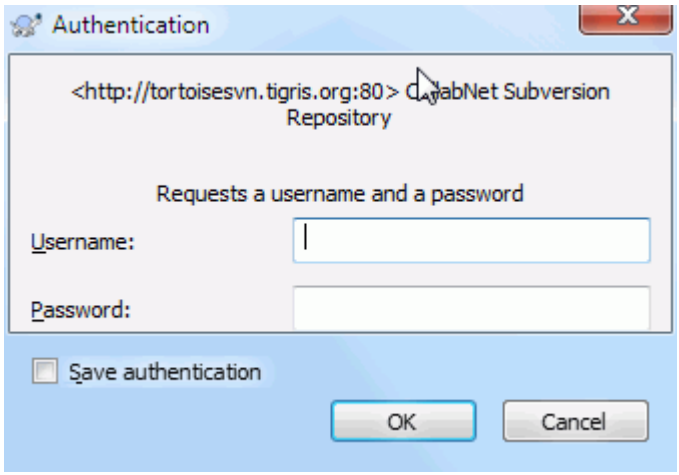
F5
: Osveži trenutni pogled. To je verjetno najbolj uporaben ukaz, ki ga lahko izvršite z eno tipko. Na primer: v Raziskovalcu boste s tem osvežili prekrivne ikone v delovni kopiji. V oknu za objave se bo s tem ponovno pregledala delovna kopija, v okno pa se bodo dodale najdene spremembe. V oknu dnevniških zapisov bo to povzročilo ponovno kontaktiranje skladišča in preverjanje, če obstajajo novejši zapisi.

Ctrl-A
: Izberi vse. To lahko uporabite, če se pojavi sporočilo o napaki in ga želite prekopirati in prilepiti v elektronsko pošto. Uporabite Ctrl-A, če želite izbrati sporočilo o napaki in potem...

Ctrl-C
: Copy the selected text. In case no text is selected but e.g. a list entry or a message box, then the content of that list entry or the message box is copied to the clipboard.

4.1.5. Avtentikacija

Če je skladišče, do katerega želite dostopati, zaščiteno z geslom, se prikaže avtentikacijsko pogovorno okno.



Slika 4.5. Okno za avtentikacijo

Vpišite svoje uporabniško ime in geslo. Potrditveno polje pove programu TortoiseSVN, naj avtentikacijske podatke shrani v privzeto mapo: `%APPDATA%\Subversion\auth` v tri podmape:

- `svn.simple` contains credentials for basic authentication (username/password). Note that passwords are stored using the WinCrypt API, not in plain text form.
- `svn.ssl.server` vsebuje certificate strežnika SSL
- `svn.username` vsebuje avtentikacijske podatke za avtentikacijo samo z uporabniškim imenom (brez gesla).

If you want to clear the authentication cache, you can do so from the Saved Data page of TortoiseSVN's settings dialog. The button Clear all will clear the cached authentication data for all repositories. The button Clear...

however will show a dialog where you can chose which cached authentication data should be deleted. Refer to Razdelek 4.31.6, “Shranjeni podatki”.

Some people like to have the authentication data deleted when they log off Windows, or on shutdown. The way to do that is to use a shutdown script to delete the `%APPDATA%\Subversion\auth` directory, e.g.

```
@echo off
rmdir /s /q "%APPDATA%\Subversion\auth"
```

You can find a description of how to install such scripts at http://www.windows-help-central.com/windows-shutdown-script.html.

For more information on how to set up your server for authentication and access control, refer to Razdelek 3.5, “Accessing the Repository”.

4.1.6. Povečevanje oken

Kar nekaj pogovornih oken TortoiseSVN prikaže veliko informacij. Pogosto je uporabno, če povečate le širino ali le višino, namesto da povečate okno na celo stran. Z lažje prilagajanje je tu gumb Razpni. Uporabite srednji miškin gumb za podaljšanje navpične stranice na višino zaslona in desni miških gumb za podaljšanje vodoravne stranice na širino zaslona.

4.2. Uvažanje podatkov v skladišče

4.2.1. Uvoz

If you are importing into an existing repository which already contains some projects, then the repository structure will already have been decided. If you are importing data into a new repository, then it is worth taking the time to think about how it will be organised. Read Razdelek 3.1.5, “Postavitev skladišča” for further advice.

This section describes the Subversion import command, which was designed for importing a directory hierarchy into the repository in one shot. Although it does the job, it has several shortcomings:

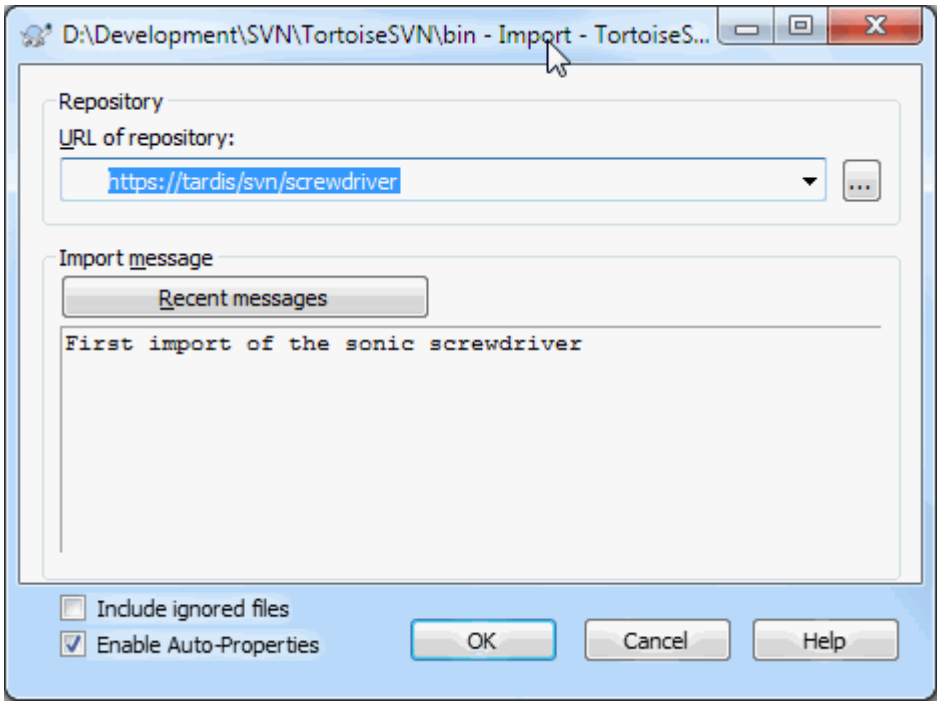
- Datotek in map, ki jih želite vključiti, ne morete izbrati, lahko pa si pomagate z nastavitvami splošnega vzorca prezrtih elemetov.
- Uvožena mapa ne postane delovna kopija. Narediti morate prevzem datotek iz strežnika.
- Povsem preprosto je uvoziti v napačno mapo znotraj hierarhije skladišča.

For these reasons we recommend that you do not use the import command at all but rather follow the two-step method described in Razdelek 4.2.2, “Uvažanje na mestu”, unless you are performing the simple step of creating an initial `/trunk /tags /branches` structure in your repository. Since you are here, this is how the basic import works ...

Preden projekt uvozite v skladišče, morate:

1. Odstraniti vse datoteke, ki niso potrebne za gradnjo projekta (začasne datoteke, datoteke, ki jih generira prevajalnik, n. pr. *.obj, prevedene dvojiške datoteke...).
2. Organizirajte datoteke v mape in podmape. Čeprav je možno preimenovanje/premikanje tudi kasneje, je zelo priporočljivo, da postavite strukturo projekta pred uvažanjem.

Sedaj izberite vrhnjo datoteko vašega projekta v Raziskovalcu in desno kliknite. S tem odprete kontekstni meni. Izberite ukaz TortoiseSVN → Uvozi..., ki prikaže naslednje okno:



Slika 4.6. Okno za uvažanje

In this dialog you have to enter the URL of the repository location where you want to import your project. It is very important to realise that the local folder you are importing does not itself appear in the repository, only its content. For example if you have a structure:

```
C:\Projects\Widget\source
C:\Projects\Widget\doc
C:\Projects\Widget\images
```

and you import `C:\Projects\Widget` into `http://mydomain.com/svn/trunk` then you may be surprised to find that your subdirectories go straight into `trunk` rather than being in a `Widget` subdirectory. You need to specify the subdirectory as part of the URL, `http://mydomain.com/svn/trunk/Widget-X`. Note that the import command will automatically create subdirectories within the repository if they do not exist.

Sporočilo uvoza se uporabi kot sporočilo dnevniškega zapisa.

Po privzetih nastavitvah se datoteke in mape, ki ustrezajo splošnemu vzorcu prezrtih elementov, *ne* uvozijo. Če želite to pri posameznem uvozu spremeniti, potrdite potrditveno polje Vključi prezrte datoteke. Preberite Razdelek 4.31.1, “Splošne nastavitve” za več informacij o nastavljanju splošnih vzorcev prezrtih elementov.

Ko pritisnete gumb V redu, TortoiseSVN uvozi celotno strukturo map, vključno z vsemi datotekami, v skladišče. Projekt je sedaj shranjen v skladišču pod nadzorom različic. Upoštevajte, da mapa, ki ste jo uvozili, *NI* pod nadzorom različic! Da bi dobili *delovno kopijo* datotek pod nadzorom, morate narediti prevzem verzije, ki ste jo pravkat uvozili. Ali pa berite naprej, da izveste, kako uvoziti mapo na mestu.

4.2.2. Uvažanje na mestu

Ob predpostavki, da skladišče že imate in želite dodati vanj novo drevesno strukturo, naredite naslednje:

1. Use the repository browser to create a new project folder directly in the repository. If you are using one of the standard layouts you will probably want to create this as a sub-folder of trunk rather than in the repository root. The repository browser shows the repository structure just like Windows explorer, so you can see how things are organised.

2. Checkout the new folder over the top of the folder you want to import. You will get a warning that the local folder is not empty. Ignore the warning. Now you have a versioned top level folder with unversioned content.

3. Uporabite TortoiseSVN → Dodaj... na mapi pod nadzorom. S tem lahko dodate delno ali celotno vsebino, dodate ali odstranite datoteke, nastavite lastnosti `svn:ignore` na mapah in naredite vse ostale spremembe, ki so potrebne.

4. Objavite vrhnjo mapo in dobili ste novo drevo pod nadzorom in krajevno delovno kopijo, ustvarjeno iz obstoječe mape.

4.2.3. Posebne datoteke

Včasih imate v sistemu različic datoteko, ki vsebuje podatke, specifične za uporabnika. To pomeni, da imate datoteko, ki jo mora vsak uporabnik urediti, da ustreza njegovi/njeni namestitvi. Imeti takšno datoteko pod nadzorom je težko, ker bi vsak uporabnik ob vsaki objavi objavil svoje spremembe te datoteke.

V takšnih primerih priporočamo uporabo *predlog*. Ustvarite datoteko, ki vsebuje vse podatke, ki jih razvijalci potrebujejo, jo dodajte v sistem nadzora različic in pustite, da jo razvijalci prevzamejo. Nato si vsak razvijalec *naredi kopijo* te datoteke in jo preimenuje. Po tem spreminjanje kopije ni več problematično.

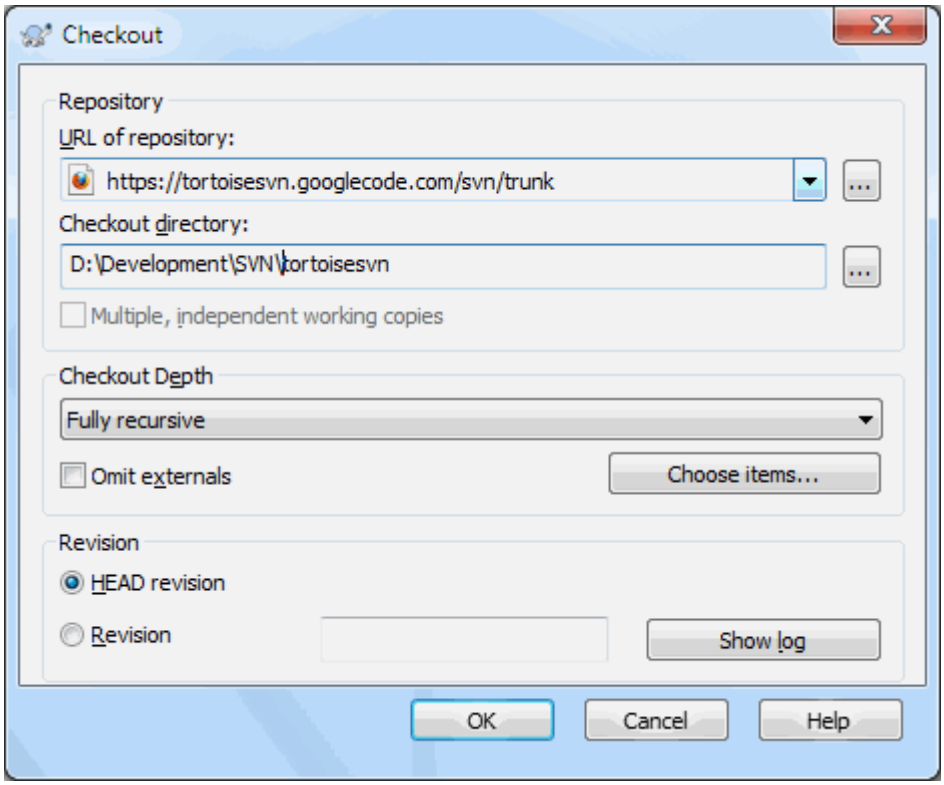
As an example, you can have a look at TortoiseSVN's build script. It calls a file named `default.build.user` which doesn't exist in the repository. Only the file `default.build.user.tmpl`. `default.build.user.tmpl` is the template file which every developer has to create a copy from and rename that file to `default.build.user`. Inside that file, we added comments so that the users will see which lines they have to edit and change according to their local setup to get it working.

So as not to disturb the users, we also added the file `default.build.user` to the ignore list of its parent folder, i.e. we've set the Subversion property `svn:ignore` to include that filename. That way it won't show up as unversioned on every commit.

4.3. Prevzemanje delovne kopije

Da bi ustvarili delovno kopijo, morate narediti *prevzem* iz skladišča.

V Raziskovalcu izberite mapo, kjer želite imeti delovno kopijo. Z desnim klikom prikličite kontekstni meni in izberite ukaz TortoiseSVN → Prevzemi..., ki ponudi naslednje pogovorno okno:



Slika 4.7. Okno za prevzem

Če vnesete ime mape, ki še ne obstaja, se le-ta pred prevzemom samodejno ustvari.

Pomembno

In the default setting, the checkout menu item is not located in the TortoiseSVN submenu but is shown at the top explorer menu. TortoiseSVN commands that are not in the submenu have SVN prepended: SVN Checkout...

4.3.1. Globina prevzema

Lahko določite tudi *globino* prevzema. Ta označuje, kako globoko v mape deluje prevzem. Če želite prevzeti le nekatere mape velikega drevesa, prevzemite le vrhnjo mapo, nato pa rekurzivno posodobite posamezne izbrane mape.

Popolnoma rekurzivno
: Prevzemi celotno drevo, vključno z mapami in podmapami.

Takojšnji nasledniki, vključno z mapami
: Prevzemi določeno mapo vključno z datotekami in podmapami, vendar brez vsebine podmap.

Samo podrejene datoteke
: Prevzemi izbrano mapo, vključno z vsemi datotekami, vendar brez podmap.

Samo ta element
: Prevzemi le mapo. Ne prenašaj vsebine (datetok in podmap).

Delovna kopija
: Ohrani globino, določeno v delovni kopiji. Ta možnost se ne uporablja v oknu za prevzem, ampak je privzeta nastavitev v vseh ostalih oknih, kjer je mogoče nastaviti globino.

Izloči
: Used to reduce working copy depth after a folder has already been populated. This option is only available in the Update to revision dialog.

To easily select only the items you want for the checkout and force the resulting working copy to keep only those items, click the Choose items... button. This opens a new dialog where you can check all items you want in your working copy and uncheck all the items you don't want. The resulting working copy is then known as a `sparse checkout`. An update of such a working copy will not fetch the missing files and folders but only update what you already have in your working copy.

If you check out a sparse working copy (i.e., by choosing something other than `fully recursive` for the checkout depth), you can easily add or remove sub-folders later using one of the following methods.

4.3.1.1. Sparse Update using Update to Revision

Right click on the checked out folder, then use TortoiseSVN → Update to Revision and select Choose items.... This opens the same dialog that was available in the original checkout and allows you to select or deselect items to include in the checkout. This method is very flexible but can be slow as every item in the folder is updated individually.

4.3.1.2. Sparse Update using Repo Browser

Right click on the checked out folder, then use TortoiseSVN → Repo-Browser to bring up the repository browser. Find the sub-folder you would like to add to your working copy, then use Context Menu → Update item to revision....

4.3.1.3. Sparse Update using Check for Modifications

In the check for modifications dialog, first **shift** click on the button Check repository. The dialog will show all the files and folders which are in the repository but which you have not checked out as `remotely added`. Right click on the folder(s) you would like to add to your working copy, then use Context menu → Update.

Ta zmožnost je uporabna, kadar želite prevzeti dele velikega drevesa, hkrati pa ohraniti udobje posodabljanja ene same delovne kopije. Denimo, da imate veliki drevo s podmapami `Projekt01` do `Projekt99`, prevzeti pa želite le mape `Projekt03`, `Projekt25` and `Projekt76/Podprojekt`. Naredite naslednje korake:

1. Prevzemite nadrejeno mapo z globino “Samo ta element”. S tem ste ustvarili prazno vrhnjo mapo.
2. Za prikaz vsebine skladišča izberite novo mapo in uporabite TortoiseSVN → Brskalnik po skladišču.
3. Desno kliknite na mapo `Projekt03` in izberite Kontekstni meni → Posodobi element na revizijo.... Uporabite prevzete nastavitve in kliknite na gumb V redu. S tem ste prevzeli vsebino mape.

   Isti postopek ponovite za mapo `Projekt25`.
4. Pojdite na `Projekt76/Podprojekt` in ponovite postopek. Mapa `Projekt76` nima vsebine - izjema je le mapa `Podprojekt`, katere vsebina je prenešena. Subversion je ustvaril vmesne mapi brez prenašanja vsebine.

Changing working copy depth

Once you have checked out a working copy to a particular depth you can change that depth later to get more or less content using Context menu → Update item to revision.... In that dialog, be sure to check the Make depth sticky checkbox.

Uporaba starejšega strežnika

Pre-1.5 servers do not understand the working copy depth request, so they cannot always deal with requests efficiently. The command will still work, but an older server may send all the data, leaving

> the client to filter out what is not required, which may mean a lot of network traffic. If possible you should upgrade your server to at least 1.5.

Če projekt vsebuje reference na zunanje projekte, ki jih *ne* želite prevzeti istočasno, uporabite potrditveno polje Izpusti zunanje.

Pomembno

Če je možnost Izpusti zunanje potrjena ali če želite povečati globino, morate narediti posodobitev delovne kopije z uporabo ukaza TortoiseSVN → Posodobi na revizijo... namesto ukaza TortoiseSVN → Posodobi. Standardna posodobitev vključuje vse zunanje projekte in obdrži obstoječo globino.

Priporočamo, da prevzamete samo `trunk` del strukture map ali manj. Če uporabite vrhnjo mapo, se vam lahko zgodi, da boste imeli povsem poln trdi disk, saj boste prevzeli prav vsako kopijo drevesne strukture, vključno z vsemi vejami in oznakami vašega projekta!

Izvažanje

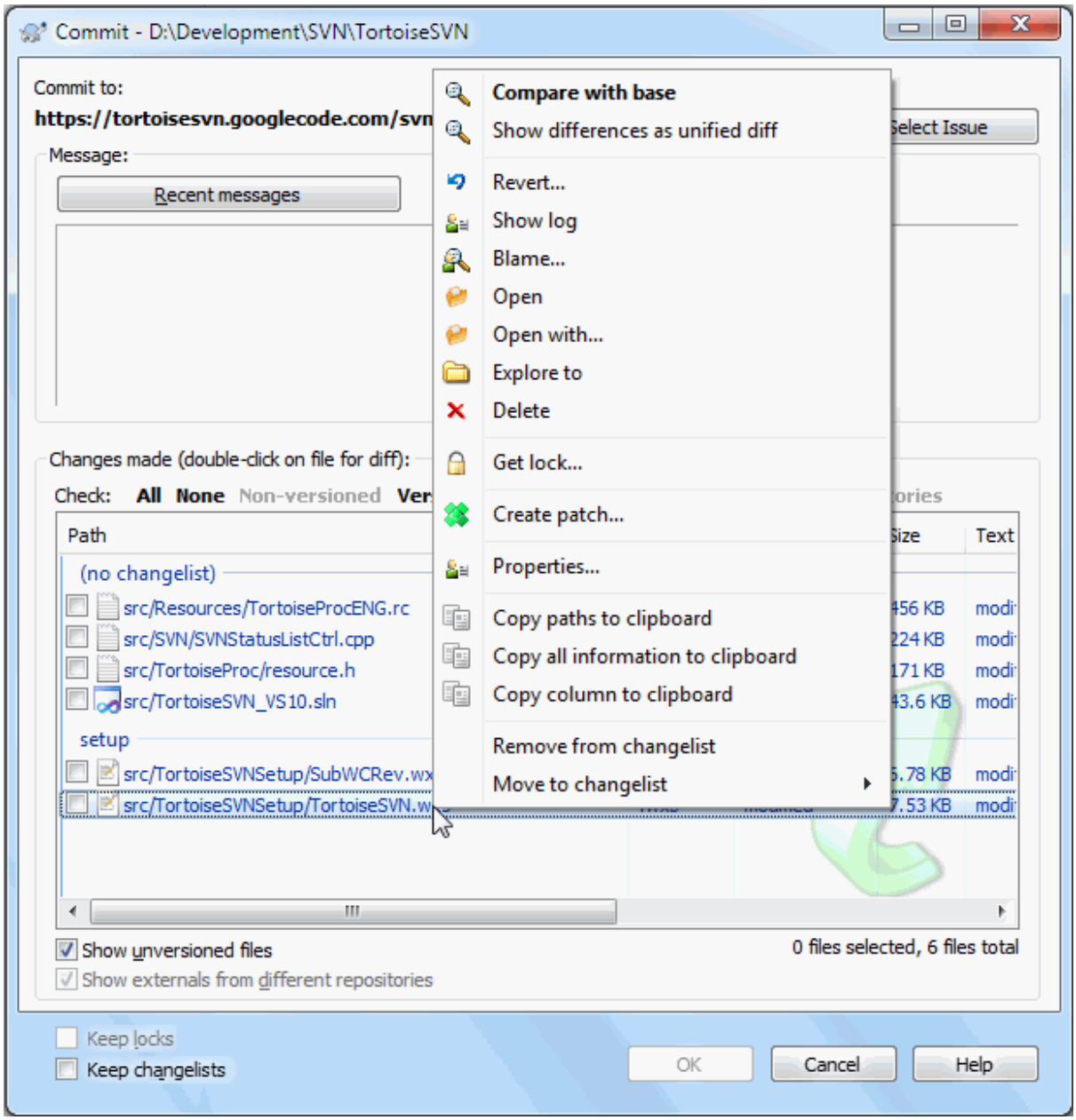
Včasih potrebujete krajevno kopijo brez map `.svn`, n. pr. če želite zapakirati svojo izvorno kodo. Več o tem si preberite v Razdelek 4.27, “Izvažanje delovne kopije sistema Subversion”.

4.4. Objavljanje sprememb v skladišču

Pošiljanje sprememb, ki ste jih naredili na delovni kopiji, se imenuje *objava* sprememb. Preden pa spremembe objavite, se morate prepričati, da je vaša delovna kopija posodobljena. Uporabite lahko ukaz TortoiseSVN → Posodobi. Lahko pa uporabite ukaz TortoiseSVN → Preveri spremembe, da vidite, katere datoteke so bile spremenjene v delovni kopiji ali v skladišču.

4.4.1. Okno objave

Če je delovna kopija sodobna in brez sporov, ste pripravljeni, da spremembe objavite. Izberite datoteke in/ali mape, ki jih želite objaviti, in izberite TortoiseSVN → Objavi....



Slika 4.8. Okno objave

Okno za objave pokaže vsako spremenjeno datoteko, vključno z dodanimi, izbrisanimi in datotekami brez različic. Če ne želite, da se spremembe določene datoteke objavijo, izklopite potrditveno polje pred to datoteko. Če želite vključiti datoteko brez različic, potrdite potrditveno polje - s tem jo dodate v objavo.

To quickly check or uncheck types of files like all versioned files or all modified files, click the link items just above the list of shown items.

For information on the coloring and overlays of the items according to their status, please see Razdelek 4.7.3, “Krajevno in oddaljeno stanje”.

Elementi, ki so bili preklopljeni na drugo pot v skladišču, so označeni z znakom `(s)`. Lahko da ste kakšen element preklopili, ko ste delali na neki veji in ga pozabili preklopiti nazaj. To je opozorilo!

Naj objavim datoteke ali mape?

Ko objavljate datoteke, pogovorno okno prikaže le datoteke, ki ste jih izbrali. Ko objavljate mapo, pogovorno okno izbere spremenjene datoteke samodejno. Če ste pozabili na novo datoteko, jo okno pri objavljanju mape vseeno najde. Objava mape *ne* pomeni, da se vse datoteke označijo kot spremenjene, ampak vam le olajša delo, saj ga opravi namesto vas.

Veliko datotek brez različic v oknu objave

Če se vam zdi, da okno za objavo prikaže preveliko število datotek brez različic (generirane datoteke razvojnega okolja, varnostne datoteke), obstaja več načinov, da situacijo popravite. Lahko:

- dodate datoteko (ali končnico) na seznam izločenih datotek, kar storite v nastavitvah. To vpliva na vse delovne kopije, ki jih imate.
- dodate datoteko na seznam prezrtih (`svn:ignore`) z uporabo ukaza TortoiseSVN → Dodaj na seznam prezrtih Ta nastavitev vpliva le na mapo, v kateri ste nastavili lastnost `svn:ignore`. Z uporabo okna Lastnosti lahko lastnost mape spremenite.
- add the file to the `svn:global-ignores` list using TortoiseSVN → Add to ignore list (recursively) This will affect the directory on which you set the `svn:global-ignores` property and all subfolders as well.

Za več informacij preberite Razdelek 4.14, “Dodajanje datotek in map na seznam prezrtih elementov”.

Dvoklik na spremenjeno datoteko v oknu objav požene zunanje orodje za razlikovanje in vam prikaže spremembe. Kontekstni meni vam ponuja še več možnosti, kar je razvidno iz posnetka zaslona. Poleg tega lahko iz okna potegnete in prenesete datoteke v ostale aplikacije, npr. v urejevalnik besedil ali ravojno okolje (IDE).

Izbiro elementov naredite s potrditvijo polja pred posameznim elementom. Za mape lahko uporabite označevanje s tipko **Shift**-izbira, s čemer naredite akcijo rekurzivno.

Stolpci v spodnjem delu okna so povsem prilagodljivi. Z desnim klikom na glavo kateregakoli stolpca prikličete kontekstni meni, ki vam omogoča izbiro stolpcev, ki jih želite prikazati. Prav tako lahko spremenite širino stolpcev. To storite tako, da rob stolpca premaknete, ko je miška na meji med dvema stolpcema. Prilagoditve se ohranijo, tako da boste naslednjič videli enako postavitev.

Če imate nastavljene privzete nastavitve in naredite objavo sprememb, se vsi zaklepi v vaši lasti sprostijo takoj po uspešno izvedeni objavi. Če želite zaklepe ohraniti, potrdite polje Ohrani zaklepe. Privzeto stanje tega polja je definirano v konfiguracijski datoteki sistema Subversion z možnostjo `no_unlock`. Za več informacij o urejanju konfiguracijske datoteke Subversion si preberite Razdelek 4.31.1, “Splošne nastavitve”.

Warning when committing to a tag

Usually, commits are done to the trunk or a branch, but not to tags. After all, a tag is considered fixed and should not change.

If a commit is attempted to a tag URL, TortoiseSVN shows a confirmation dialog first to ensure whether this is really what is intended. Because most of the time such a commit is done by accident.

However, this check only works if the repository layout is one of the recommended ones, meaning it uses the names `trunk`, `branches` and `tags` to mark the three main areas. In case the setup is different, the detection of what is a tag/branch/trunk (also known as `classification patterns`), can be configured in the settings dialog: Razdelek 4.31.2, “Revision Graph Settings”

Povleci in spusti

Datoteke lahko povlečete in spustite v okno za objavo od drugod pod pogojem, da ste delovne kopije prevzeli iz istega skladišča. Primer: imato ogromno delovno kopijo in uporabljate več Raziskovalcev, ki ima nastavljene različne trenutne mape znotraj delovne kopije. Če se želite izogniti objavljanju vrhnje mape (kar bi pomenilo dolgo iskanje spremenjenih datotek), lahko okno za objave odprete v neki mapi, nato pa iz različnih Raziskovalcev povlečete v okno datoteke, ki jih želite istočasno objaviti.

Datoteke brez različic lahko povlečete v pogovorno okno za objavo in bodo samodejno dodane v sistem Subversion.

Dragging files from the list at the bottom of the commit dialog to the log message edit box will insert the paths as plain text into that edit box. This is useful if you want to write commit log messages that include the paths that are affected by the commit.

Popravljanje zunanjih preimenovanj

Včasih datoteke preimenujete zunaj sistema Subversion, zato se v seznamu sprememb pojavita manjkajoča datoteka in datoteka brez različic. Da bi preprečili izgubo zgodovine, morate sistemu Subversion povedati, da med datotekama obstaja povezava. Enostavno izberite tako staro ime (manjkajoče) kot novo ime (brez različic) in uporabite Kontekstni meni → Popravi premik in s tem povežete datoteki v preimenovanje.

Repairing External Copies

If you made a copy of a file but forgot to use the Subversion command to do so, you can repair that copy so the new file doesn't lose its history. Simply select both the old name (normal or modified) and the new name (unversioned) and use Context Menu → Repair Copy to pair the two files as a copy.

4.4.2. Seznami sprememb

Okno za objave omogoča uporabo seznamov sprememb, kar pomaga pri združevanju sprememb v logične skupine. Za več informacij o tej zmožnosti preberite Razdelek 4.8, “Seznami sprememb”.

4.4.3. Commit only parts of files

Sometimes you want to only commit parts of the changes you made to a file. Such a situation usually happens when you're working on something but then an urgent fix needs to be committed, and that fix happens to be in the same file you're working on.

right click on the file and use Context Menu → Restore after commit. This will create a copy of the file as it is. Then you can edit the file, e.g. in a text editor and undo all the changes you don't want to commit. After saving those changes you can commit the file.

Uporaba TortoiseMerge

If you use TortoiseMerge to edit the file, you can either edit the changes as you're used to, or mark all the changes that you want to include. right click on a modified block and use Context Menu → Mark this change to include that change. Finally right click and use Context Menu → Leave only marked changes which will change the right view to only include the changes you've marked before and undo the changes you have not marked.

After the commit is done, the copy of the file is restored automatically, and you have the file with all your modifications that were not committed back.

4.4.4. Izključevanje elementov iz okna objav

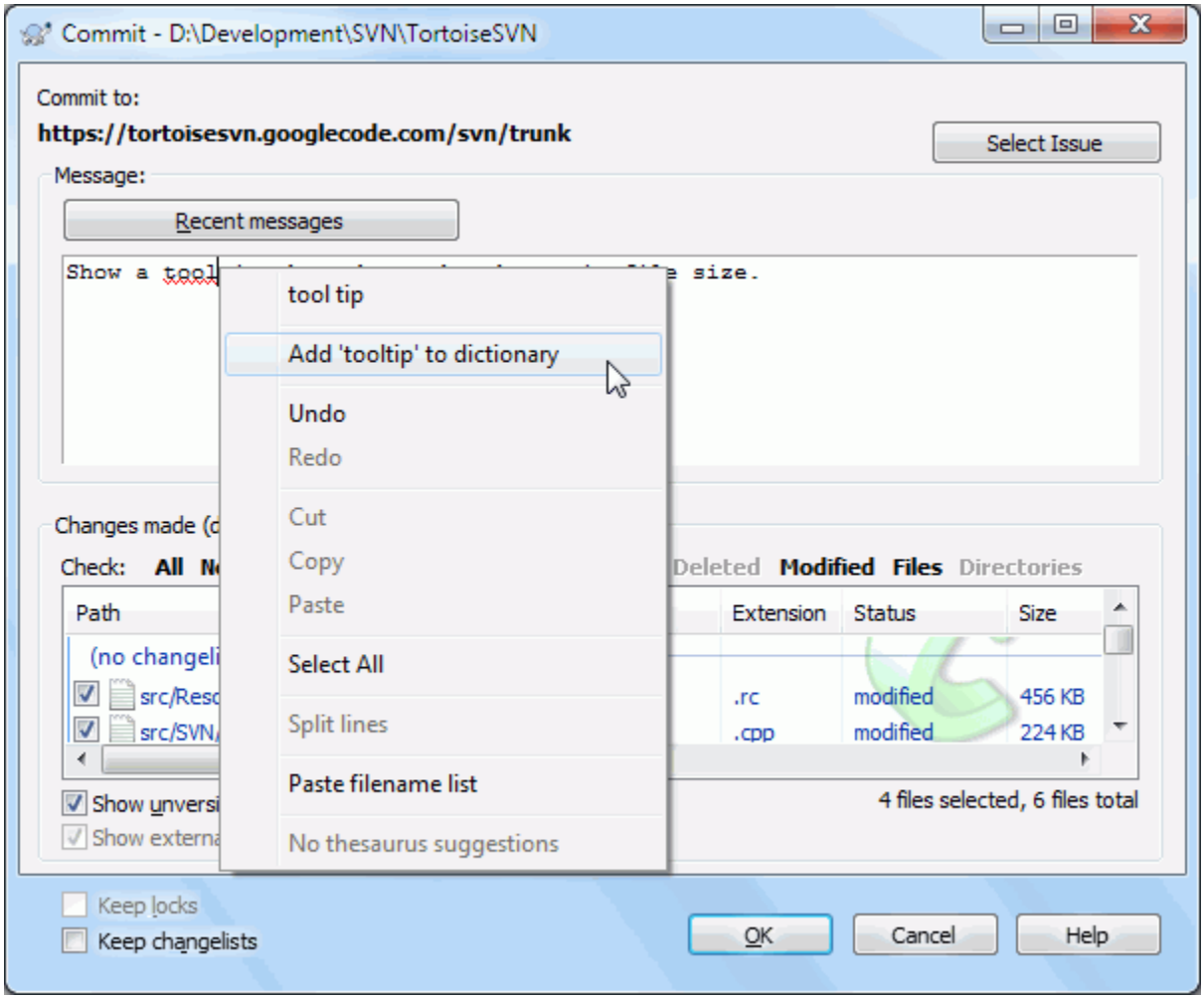
Včasih se v skladišču nahajajo datoteke, ki se pogosto spreminjajo, vendar jih ne želite vedno znova objavljati. Včasih to pomeni napako v procesu gradnje projekta - zakaj so te datoteke sploh pod nadzorom različic? Mogoče bi morali uporabljati predloge? Včasih pa je takšna situacija neizogibna. Primer: razvojno okolje spremeni časovni žig datoteke pri vsaki gradnji. Projektna datoteka mora biti pod nadzorom, saj vsebuje nastavitve za gradnjo, vendar ni potrebe, da jo objavljamo ob vsaki spremembi časovnega žiga.

Da bi olajšali delo v tašnih primerih, smo izdelali poseben seznam sprememb, ki se imenuje `ignore-on-commit`. Datoteke na tem seznamu so samodejno izključene iz izbora v oknu objav. Spremembe lahko še vedno objavite, vendar morate ročno izbrati takšno datoteko.

4.4.5. Sporočila dnevniških zapisov objav

Vedno vnesite sporočilo dnevniškega zapisa, ki opisuje spremembe, ki jih objavljate. S tem pri pregledovanju dnevniških zapisov vidite, kaj se je spremenilo in kdaj. Sporočilo je lahko poljubne dolžine. Veliko projektov natančno določa, kaj je potrebno vpisati, v katerem jeziku, včasih pa je določena zelo natančna oblika zapisa.

Sporočilom dnevniških zapisov lahko dodate enostavno oblikovanje, podobno tistemu, ki ga uporabljate pri elektronski pošti. Če želite `besedilu` dodati oblikovanje, uporabite `*besedilo*` za polkrepko pisavo, `_besedilo_` za podčrtano pisavo in `^besedilo^` za poševno pisavo.



Slika 4.9. Črkovalnik v oknu objave

TortoiseSVN vsebuje črkovalnik, ki vam pomaga pri pravilnem pisanju sporočil dnevniških zapisov. Označil bo vse nepravilno črkovane besede. Uporabite kontekstni meni za dostop do predlaganih popravkov. Seveda ne pozna

prav *vseh* tehničnih terminov, tako bodo pravilno črkovane besede včasih označene kot napačne. Ne skrbite. Z uporabo kontekstnega menija jih lahko dodate v osebni slovar.

The log message window also includes a filename and function auto-completion facility. This uses regular expressions to extract class and function names from the (text) files you are committing, as well as the filenames themselves. If a word you are typing matches anything in the list (after you have typed at least 3 characters, or pressed **Ctrl+Space**), a drop-down appears allowing you to select the full name. The regular expressions supplied with TortoiseSVN are held in the TortoiseSVN installation `bin` folder. You can also define your own regexes and store them in `%APPDATA%\TortoiseSVN\autolist.txt`. Of course your private autolist will not be overwritten when you update your installation of TortoiseSVN. If you are unfamiliar with regular expressions, take a look at the introduction at https://en.wikipedia.org/wiki/Regular_expression, and the online documentation and tutorial at http://www.regular-expressions.info/.

Getting the regex just right can be tricky, so to help you sort out a suitable expression there is a test dialog which allows you to enter an expression and then type in filenames to test it against. Start it from the command prompt using the command `TortoiseProc.exe /command:autotexttest`.

The log message window also includes a commit message snippet facility. These snippets are shown in the autocomplete dropdown once you type a snippet shortcut, and selecting the snippet in the autocomplete dropdown then inserts the full text of the snippet. The snippets supplied with TortoiseSVN are held in the TortoiseSVN installation `bin` folder. You can also define your own snippets and store them in `%APPDATA%\TortoiseSVN\snippet.txt`. `#` is the comment character. Newlines can be inserted by escaping them like this: `\n` and `\r`. To insert a backslash, escape it like this: `\\`.

Predhodno vnesena sporočila dnevniških zapisov lahko ponovno uporabite. Enostavno kliknite na Zadnja sporočila in prikaže se vam nekaj zadnjih sporočil, ki ste jih vnesli v tej delovni kopiji. Število shranjenih sporočil lahko nastavite v oknu za nastavitve TortoiseSVN.

Shranjena sporočila lahko izbrišete na strani Shranjeni podatki v oknu za nastavitve TortoiseSVN, posamezna sporočila pa lahko izbrišete iz okna Zadnja sporočila, če držite pritisnjeno tipko **Delete**.

If you want to include the checked paths in your log message, you can use the command Context Menu → Paste filename list in the edit control.

Another way to insert the paths into the log message is to simply drag the files from the file list onto the edit control.

Posebne lastnosti map

Obstaja več posebnih lastnosti za mape, ki vam omogočajo več nadzora nad oblikovanjem sporočil dnevniških zapisov in jezikom, ki se uporabi za preverjanje črkovanja. Več o tem pove Razdelek 4.18, “Nastavitve projekta”.

Integracija s sistemi za sledenje hroščev

Če uporabljate sistem za sledenje zadev, lahko v vnosno polje ID hrošča / št. zadeve: vpišete eno ali več številk zadev. Če vpišete več številk, naj bodo ločene z vejico. Druga možnost je, da uporabljate podporo sistem za sledenje zadev na osnovi regularnih izrazov. V tem primeru lahko številko zadeve navedete nekje znotraj sporočila dnevniškega zapisa. Več o tem vam pove Razdelek 4.29, “Integracija s sistemi za sledenje zadev”.

4.4.6. Napredek objave

Po pritisku na gumb V redu se pojavi okno napredka objave.

Slika 4.10. Okno napredka prikazuje napredovanje objave

Okno napredka uporablja barvno kodiranje za označevanje različnih operacij objave.

Modra
: Objavljanje spremembe.

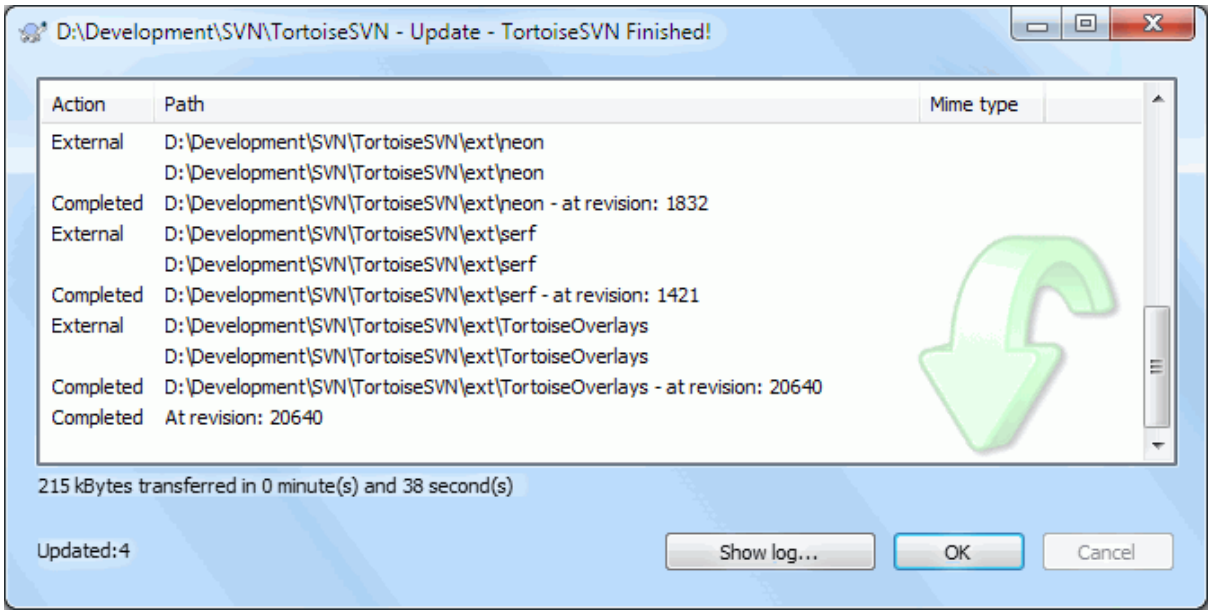
Škrlatna
: Objavljanje novega elementa.

Temno rdeča
: Objavljanje brisanja ali zamenjave.

Črna
: Ostali elementi.

To je privzeta barvna shema, vendar lahko barve poljubno nastavite v oknu za nastavitve. Za več informacij preberite Razdelek 4.31.1.5, “Nastavitev barv TortoiseSVN”.

4.5. Posodobite delovno kopijo s spremembami ostalih uporabnikov



Slika 4.11. Okno napredka prikazuje končano posodobitev

Ob določenih časovnih razmakih morate poskrbeti, da se spremembe, ki jih na projektu naredijo drugi uporabniki, vnesejo v vašo delovno kopijo. Procesu prenosa sprememb iz strežnika v delovno kopijo rečemo *posodabljanje*. Posodobite lahko posamezne datoteke, skupino izbranih datotek ali celotno hierarhijo map. Ko želite izvesti posodobitev, izberite datoteke in/ali mape in iz kontekstnga menija zaženite TortoiseSVN → Posodobi. Pojavi se okno, ki prikazuje napredek posodobitve. Spremembe, ki so jih naredili ostali uporabniki, se spojijo v vašo delovno kopijo, pri tem pa se vaše spremembe ohranijo. V skladišču se s posodobitvijo delovne kopije *ne* zgodi nobena sprememba.

Okno napredka uporablja barvno označevanje različnih akcij posodobitev

Škrlatna
: Nov element dodan v delovno kopijo.

Temno rdeča
: Odvečen element brisan iz delovne kopije ali manjkajoč element zamenjan v delovni kopiji.

Zelena
: Spremembe iz skladišča so bile uspešno spojene s krajevnimi spremembami.

Svetlo rdeča
: Spremembe iz skladišča, spojene s krajevnimi spremembami, so povzročile spore, ki jih morate rešiti.

Črna
: Nespremenjem element v delovni kopiji, posodobljen iz nove verzije v skladišču.

To je privzeta barvna shema, vendar lahko barve poljubno nastavite v oknu za nastavitve. Za več informacij preberite Razdelek 4.31.1.5, “Nastavitev barv TortoiseSVN”.

Če med posodabljanjem nastanejo *spori* (to se lahko zgodi, če je nek drug uporabnik spremenil iste vrstice v datoteki kot vi, vendar so spremembe različne od vaših), okno pokaže spore v rdeči barvi. Z dvoklikom na te vrtsice zaženete zunanje orodje za spajanje, s katerim lahko spore rešite.

Ko je posodobitev končana, dialog napredka pod seznamom datotek pokaže povzetek - število posodobljenih, dodanih, odstranjenih, spornih... elementov. Povzetek lahko skopirate na odložišče s kombinacijo tipk **Ctrl+C**.

The standard Update command has no options and just updates your working copy to the HEAD revision of the repository, which is the most common use case. If you want more control over the update process, you should use TortoiseSVN → Update to Revision... instead. This allows you to update your working copy to a specific

revision, not only to the most recent one. Suppose your working copy is at revision 100, but you want it to reflect the state which it had in revision 50 - then simply update to revision 50.

In the same dialog you can also choose the *depth* at which to update the current folder. The terms used are described in Razdelek 4.3.1, “Globina prevzema”. The default depth is Working copy, which preserves the existing depth setting. You can also set the depth `sticky` which means subsequent updates will use that new depth, i.e. that depth is then used as the default depth.

To make it easier to include or exclude specific items from the checkout click the Choose items... button. This opens a new dialog where you can check all items you want in your working copy and uncheck all the items you don't want.

You can also choose whether to ignore any external projects in the update (i.e. projects referenced using `svn:externals`).

Opozorilo

Če datoteko ali mapo posodobite na neko določeno revizijo, na teh elementih ne smete delati sprememb. Če boste spremembe poskusili objaviti, boste dobili obvestilo, da so datoteke oziroma mape “zastarele”! Če želite razveljaviti spremembe na datoteki in začeti znova iz zgodnejše revizije, se lahko zavrtite nazaj na prejšnjo revizijo iz dnevnika. Preberite Razdelek B.4, “Kako prevrtim nazaj revizije v skladišču” za nadaljnje informacije in druge metode.

Update to Revision can occasionally be useful to see what your project looked like at some earlier point in its history. But in general, updating individual files to an earlier revision is not a good idea as it leaves your working copy in an inconsistent state. If the file you are updating has changed name, you may even find that the file just disappears from your working copy because no file of that name existed in the earlier revision. You should also note that the item will show a normal green overlay, so it is indistinguishable from files which are up-to-date.

If you simply want a local copy of an old version of a file it is better to use the Context Menu → Save revision to... command from the log dialog for that file.

Več datotek/map

Če v Raziskovalcu izberete več datotek in map in potem uporabite ukaz Posodobi, bodo vse datoteke/mape posodobljene ena za drugo. TortoiseSVN poskrbi, da se vse datoteke/mape, ki se nahajajo v istem skladišču, posodobijo na natančno isto revizijo, tudi če se med procesom posodabljanja v skladišču zgodi nova objava!

4.6. Reševanje sporov

Once in a while, you will get a *conflict* when you update/merge your files from the repository or when you switch your working copy to a different URL. There are two kinds of conflicts:

file conflicts
: A file conflict occurs if two (or more) developers have changed the same few lines of a file.

tree conflicts
: A tree conflict occurs when a developer moved/renamed/deleted a file or folder, which another developer either also has moved/renamed/deleted or just modified.

4.6.1. File Conflicts

A file conflict occurs when two or more developers have changed the same few lines of a file. As Subversion knows nothing of your project, it leaves resolving the conflicts to the developers. The conflicting area in a text file is marked like this:

```
<<<<<<< filename
```

```
your changes
=======
code merged from repository
>>>>>>> revision
```

Also, for every conflicted file Subversion places three additional files in your directory:

imedatoteke.kon.mine
: To je vaša datoteka v delovni kopiji, preden ste jo posodobili - ne vsebuje oznak sporov. Datoteka vsebuje zgolj vaše zadnje spremembe.

imedatoteke.kon.rSTARAREVIZIJA
: To je datoteka prevzete revizije (BASE), preden ste posodobili delovno kopijo. Torej datoteka, ki ste jo prevzeli, preden ste naredili zadnje spremembe.

imedatoteke.kon.rNOVAREVIZIJA
: To je datoteka, ki jo je odjemalec Subversion pri posodabljanju delovne kopije prejel od strežnika. Ta datoteka ustreza reviziji HEAD v skladišču.

You can either launch an external merge tool / conflict editor with TortoiseSVN → Edit Conflicts or you can use any text editor to resolve the conflict manually. You should decide what the code should look like, do the necessary changes and save the file. Using a merge tool such as TortoiseMerge or one of the other popular tools is generally the easier option as they generally present the files involved in a 3-pane view and you don't have to worry about the conflict markers. If you do use a text editor then you should search for lines starting with the string <<<<<<<.

Nato uporabite ukaz TortoiseSVN → Rešen in objavite svoje spremembe v skladišču. Upoštevajte, da ukaz Reši ne reši sporov, ampak le odstrani datoteki `imedatoteke.kon.mine` in `imedatoteke.kon.r*`, da lahko spremembe objavite.

Če se pojavijo spori pri dvojiških datotekah, Subversion ne poskuša sam spajati datotek. Krajevna datoteka ostane nespremenjena (natančno takšna kot ob vaši zadnji spremembi), pojavijo pa se datoteke `imedatoteke.kon.r*`. Če želite preklicati svoje spremembe in uporabiti različico datoteke iz skladišča, uporabite ukaz Povrni. Če želite obdržati vašo različico in povoziti različico v skladišču, uporabite ukaz Rešen, nato pa objavite spremembe.

Ukaz Rešen lahko uporabite tudi za več datotek, če kliknete na nadrejeno mapo in izberete TortoiseSVN → Rešen... Pokaže se okno s seznamom spornih datotek v mapi, v katerem izberete datoteke, ki jih želite označiti kot rešene.

4.6.2. Property Conflicts

A property conflict occurs when two or more developers have changed the same property. As with file content, resolving the conflict can only be done by the developers.

If one of the changes must override the other then choose the option to Resolve using local property or Resolve using remote property. If the changes must be merged then select Manually edit property, sort out what the property value should be and mark as resolved.

4.6.3. Tree Conflicts

A tree conflict occurs when a developer moved/renamed/deleted a file or folder, which another developer either also has moved/renamed/deleted or just modified. There are many different situations that can result in a tree conflict, and all of them require different steps to resolve the conflict.

When a file is deleted locally in Subversion, the file is also deleted from the local file system, so even if it is part of a tree conflict it cannot show a conflicted overlay and you cannot right click on it to resolve the conflict. Use the Check for Modifications dialog instead to access the Edit conflicts option.

TortoiseSVN can help find the right place to merge changes, but there may be additional work required to sort out the conflicts. Remember that after an update the working BASE will always contain the revision of each item as

it was in the repository at the time of update. If you revert a change after updating it goes back to the repository state, not to the way it was when you started making your own local changes.

4.6.3.1. Local delete, incoming edit upon update

1. Developer A modifies `Foo.c` and commits it to the repository.
2. Developer B has simultaneously moved `Foo.c` to `Bar.c` in his working copy, or simply deleted `Foo.c` or its parent folder.

An update of developer B's working copy results in a tree conflict:

- `Foo.c` has been deleted from working copy, but is marked with a tree conflict.
- If the conflict results from a rename rather than a delete then `Bar.c` is marked as added, but does not contain developer A's modifications.

Developer B now has to choose whether to keep Developer A's changes. In the case of a file rename, he can merge the changes to `Foo.c` into the renamed file `Bar.c`. For simple file or directory deletions he can choose to keep the item with Developer A's changes and discard the deletion. Or, by marking the conflict as resolved without doing anything he effectively discards Developer A's changes.

The conflict edit dialog offers to merge changes if it can find the original file of the renamed `Bar.c`. If there are multiple files that are possible move sources, then a button for each of these files is shown which allow you to chose the correct file.

4.6.3.2. Local edit, incoming delete upon update

1. Developer A moves `Foo.c` to `Bar.c` and commits it to the repository.
2. Developer B modifies `Foo.c` in his working copy.

Or in the case of a folder move ...

1. Developer A moves parent folder `FooFolder` to `BarFolder` and commits it to the repository.
2. Developer B modifies `Foo.c` in his working copy.

An update of developer B's working copy results in a tree conflict. For a simple file conflict:

- `Bar.c` is added to the working copy as a normal file.
- `Foo.c` is marked as added (with history) and has a tree conflict.

For a folder conflict:

- `BarFolder` is added to the working copy as a normal folder.
- `FooFolder` is marked as added (with history) and has a tree conflict.

  `Foo.c` is marked as modified.

Developer B now has to decide whether to go with developer A's reorganisation and merge her changes into the corresponding file in the new structure, or simply revert A's changes and keep the local file.

To merge her local changes with the reshuffle, Developer B must first find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog. Then use the button which shows the correct source file to resolve the conflict.

If Developer B decides that A's changes were wrong then she must choose the Mark as resolved button in the conflict editor dialog. This marks the conflicted file/folder as resolved, but Developer A's changes need to be removed by hand. Again the log dialog helps to track down what was moved.

4.6.3.3. Local delete, incoming delete upon update

1. Developer A moves `Foo.c` to `Bar.c` and commits it to the repository.

2. Developer B moves `Foo.c` to `Bix.c`.

An update of developer B's working copy results in a tree conflict:

- `Bix.c` is marked as added with history.
- `Bar.c` is added to the working copy with status 'normal'.
- `Foo.c` is marked as deleted and has a tree conflict.

To resolve this conflict, Developer B has to find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog.

Then developer B has to decide which new filename of `Foo.c` to keep - the one done by developer A or the rename done by himself.

After developer B has manually resolved the conflict, the tree conflict has to be marked as resolved with the button in the conflict editor dialog.

4.6.3.4. Local missing, incoming edit upon merge

1. Developer A working on trunk modifies `Foo.c` and commits it to the repository
2. Developer B working on a branch moves `Foo.c` to `Bar.c` and commits it to the repository

A merge of developer A's trunk changes to developer B's branch working copy results in a tree conflict:

- `Bar.c` is already in the working copy with status 'normal'.
- `Foo.c` is marked as missing with a tree conflict.

To resolve this conflict, Developer B has to mark the file as resolved in the conflict editor dialog, which will remove it from the conflict list. She then has to decide whether to copy the missing file `Foo.c` from the repository to the working copy, whether to merge Developer A's changes to `Foo.c` into the renamed `Bar.c` or whether to ignore the changes by marking the conflict as resolved and doing nothing else.

Note that if you copy the missing file from the repository and then mark as resolved, your copy will be removed again. You have to resolve the conflict first.

4.6.3.5. Local edit, incoming delete upon merge

1. Developer A working on trunk moves `Foo.c` to `Bar.c` and commits it to the repository.
2. Developer B working on a branch modifies `Foo.c` and commits it to the repository.

1. Developer A working on trunk moves parent folder `FooFolder` to `BarFolder` and commits it to the repository.
2. Developer B working on a branch modifies `Foo.c` in her working copy.

A merge of developer A's trunk changes to developer B's branch working copy results in a tree conflict:

- `Bar.c` is marked as added.
- `Foo.c` is marked as modified with a tree conflict.

Developer B now has to decide whether to go with developer A's reorganisation and merge her changes into the corresponding file in the new structure, or simply revert A's changes and keep the local file.

To merge her local changes with the reshuffle, Developer B must first find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog for the merge source. The conflict editor only shows the log for the working copy as it does not know which path was used in the merge, so you will have to find that yourself. The changes must then be merged by hand as there is currently no way to automate or even simplify this process. Once the changes have been ported across, the conflicted path is redundant and can be deleted.

If Developer B decides that A's changes were wrong then she must choose the Mark as resolved button in the conflict editor dialog. This marks the conflicted file/folder as resolved, but Developer A's changes need to be removed by hand. Again the log dialog for the merge source helps to track down what was moved.

4.6.3.6. Local delete, incoming delete upon merge

1. Developer A working on trunk moves `Foo.c` to `Bar.c` and commits it to the repository.
2. Developer B working on a branch moves `Foo.c` to `Bix.c` and commits it to the repository.

A merge of developer A's trunk changes to developer B's branch working copy results in a tree conflict:

- `Bix.c` is marked with normal (unmodified) status.
- `Bar.c` is marked as added with history.
- `Foo.c` is marked as missing and has a tree conflict.

To resolve this conflict, Developer B has to find out to what filename the conflicted file `Foo.c` was renamed/moved in the repository. This can be done by using the log dialog for the merge source.

Then developer B has to decide which new filename of `Foo.c` to keep - the one done by developer A or the rename done by himself.

After developer B has manually resolved the conflict, the tree conflict has to be marked as resolved with the button in the conflict editor dialog.

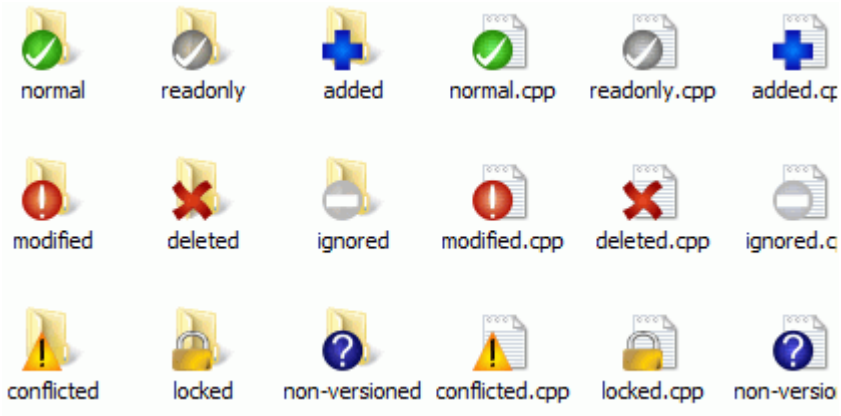
4.6.3.7. Other tree conflicts

There are other cases which are labelled as tree conflicts simply because the conflict involves a folder rather than a file. For example if you add a folder with the same name to both trunk and branch and then try to merge you will get a tree conflict. If you want to keep the folder from the merge target, just mark the conflict as resolved. If you want to use the one in the merge source then you need to SVN delete the one in the target first and run the merge again. If you need anything more complicated then you have to resolve manually.

4.7. Pridobivanje informacije o stanju

Ko delate na delovni kopiji, pogosto želite vedeti, katere datoteke ste spremenili/dodali/odstranili ali preimenovali oziroma celo, katere datoteke so spremenili in objavili ostali uporabniki.

4.7.1. Prekrivne ikone



Slika 4.12. Raziskovalec prikaže prekrivne ikone

Ko prevzamete delovno kopijo iz skladišča Subversion, imajo datoteke v Raziskovalcu spremenjene ikone. To je razlog za veliko popularnost programa TortoiseSVN. TortoiseSVN doda datotekam prekrivno ikono, ki se prekriva z originalno ikono. Prekrivna ikona je odvisna od statusa datoteke.

Sveže prevzeta delovna kopija ima zeleno prekrivno ikono. To pomeni, da je status datoteke/mape *običajno.*

Ko začnete datoteko urejati, se njen status spremeni v *spremenjeno*, prekrivna ikona pa se spremeni v rdeč klicaj. Tako hitro vidite, katere datoteke so se od zadnje posodobitve spremenile in jih je potrebno objaviti.

Če se pri posodabljanju pojavi *spor*, se ikona spremeni v rumen klicaj.

Če ste datoteki nastavili lastnost `svn:needs-lock`, Subversion napravi to datoteko samo za branje, dokler za datoteko ne pridobite zaklepa. Datoteke, ki so namenjene samo za branje, imajo takšno ikono, da lahko hitro vidite, da morate pred urejanjem datoteke zanjo pridobiti zaklep.

Če ste lastnik zaklepa datoteke, ki ima stanje *običajno*, vas ta ikona opominja, da morate zaklep sprostiti, če ga ne uporabljate, da lahko tudi drugi uporabniki objavijo spremembe na tej datoteki.

Ta ikona pove, da so nekatere datoteke ali mape znotraj trenutne mape označene *za brisanje* iz nadzora različic ali pa da se datoteka, ki je pod nadzorom, ne nahaja v mapi.

Znak plus pomeni, da je datoteka ali mapa označena *za dodajanje* v nadzor različic.

Znak pomeni, da je datoteka ali mapa v nadzoru različic *prezrta*. Prikaz te prekrivne ikone je možno vključiti ali izključiti.

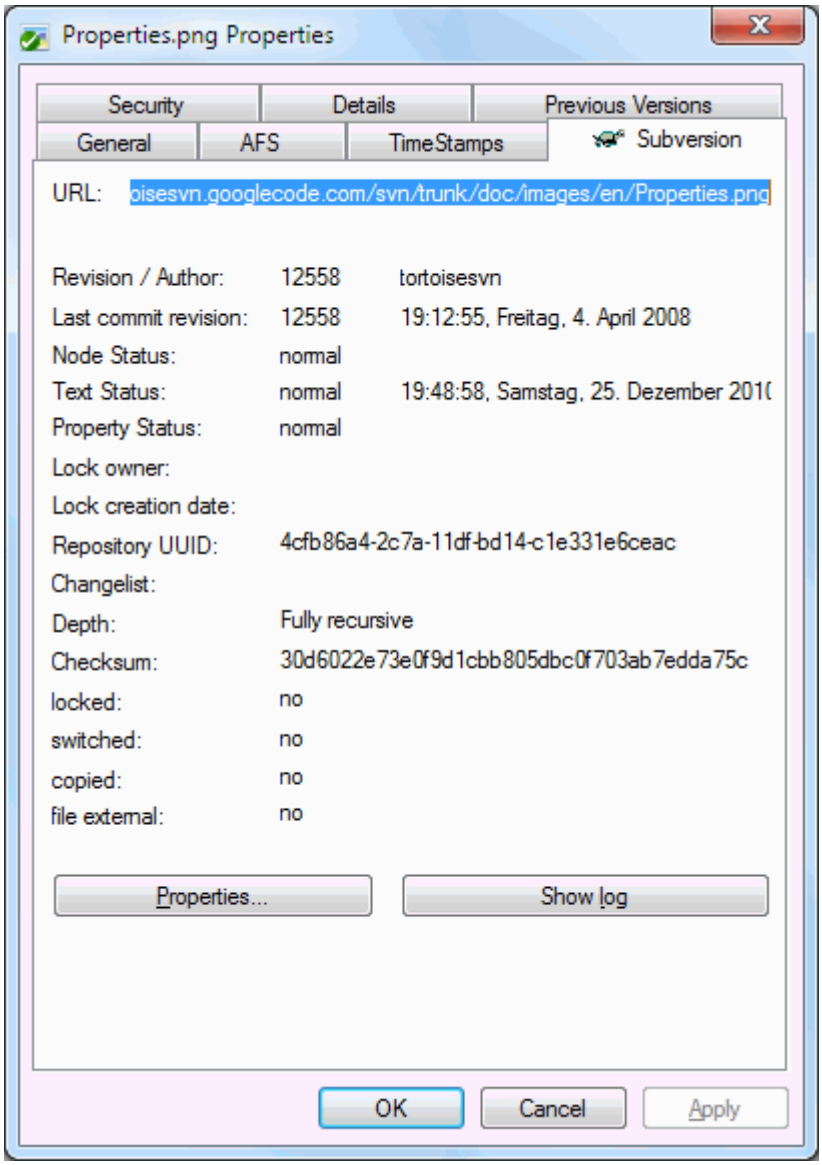
Ta ikona označuje datoteke in mape, ki niso pod nadzorom, prav tako pa niso prezrte. Uporabo te ikone lahko izključite.

Lahko celo ugotovite, da omenjene ikone niso vse na voljo na vašem sistemu. Razlog je v omejitvi števila prekrivnih ikon sistema Windows in če uporabljate tudi starejšo različico programa TortoiseCVS, potem ni več dovolj prostih prekrivnih ikon. TortoiseSVN poskuša biti “pošten državljan (TM)” in ne uporablja veliko prekrivnih ikon, da ostane prostor tudi za ostale aplikacije.

Now that there are more Tortoise clients around (TortoiseCVS, TortoiseHg, ...) the icon limit becomes a real problem. To work around this, the TortoiseSVN project introduced a common shared icon set, loaded as a DLL, which can be used by all Tortoise clients. Check with your client provider to see if this has been integrated yet :-)

Za opis, kakšnemu statusu v sistemu Subversion ustrezajo prekrivne ikone in ostale tehnične podrobnosti preberite Razdelek F.1, “Prekrivne ikone”.

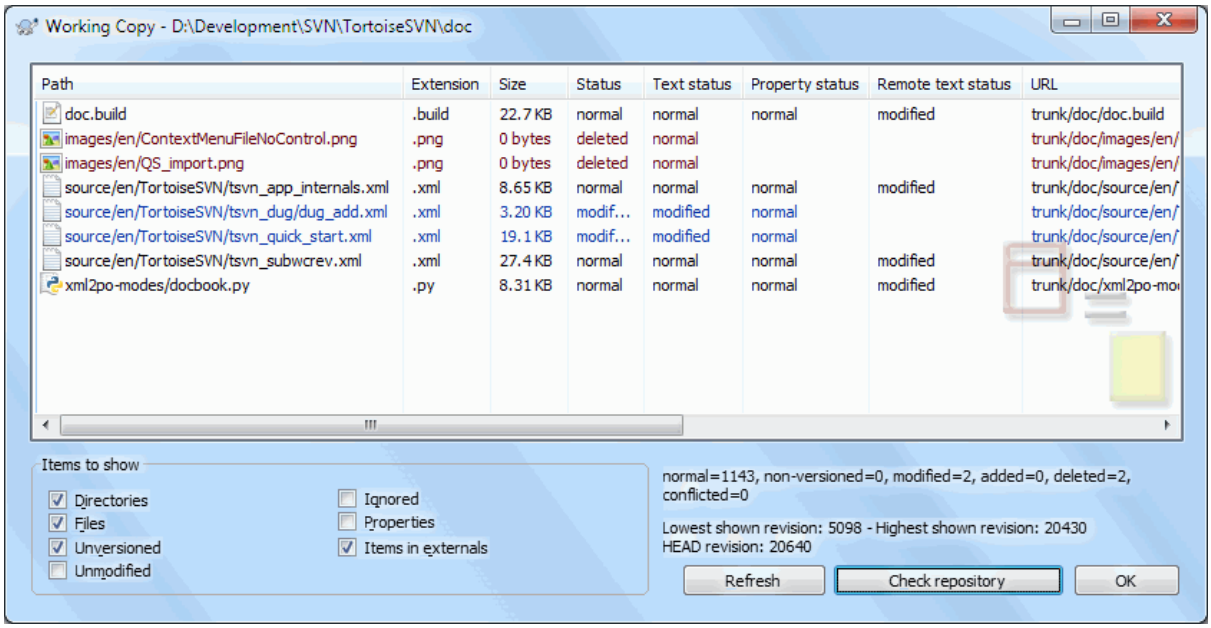
4.7.2. Detailed Status



Slika 4.13. Stran Lastnosti v Raziskovalcu, zavihek Subversion

Včasih hočete o datoteki/mapi izvedeti več, kot pove prekrivna ikona. Vse informacije, ki jih Subversion poda, lahko najdete v oknu Lastnosti. V Raziskovalcu izberite datoteko in v kontekstnem meniju izberite Meni Windows → Lastnosti (pazite: to je običajno okno Lastnosti, ki ga prikaže Raziskovalec, ne tisti v podmeniju TortoiseSVN!). V to okno za datoteke/mape pod nadzorom različic TortoiseSVN doda zavihek Subversion, kjer dobite vse potrebne informacije o izbrani datoteki/mapi.

4.7.3. Krajevno in oddaljeno stanje



Slika 4.14. Preveri posodobitve

Pogosto je zelo koristno, če veste, katere datoteke ste spremenili in katere datoteke so spremenili vaši sodelavci. Takrat pride prav ukaz TortoiseSVN → Preveri posodobitve.... To pogovorno okno prikaže vse datoteke, ki so se kakorkoli spremenile, prav tako pa tudi vse datoteke brez različic.

If you click on the Check Repository then you can also look for changes in the repository. That way you can check before an update if there's a possible conflict. You can also update selected files from the repository without updating the whole folder. By default, the Check Repository button only fetches the remote status with the checkout depth of the working copy. If you want to see all files and folders in the repository, even those you have not checked out, then you have to hold down the **Shift** key when you click on the Check Repository button.

Pogovorno okno uporablja barvno kodiranje, da poudari stanje.

Modra
: Krajevno spremenjeni elementi.

Škrlatna
: Dodani elementi. Elementi, ki so bili dodani z zgodovino, imajo pred v stolpcu Stanje besedila znak +, namig pa pokaže, od kje je bil element skopiran.

Temno rdeča
: Izbrisani ali manjkajoči elementi.

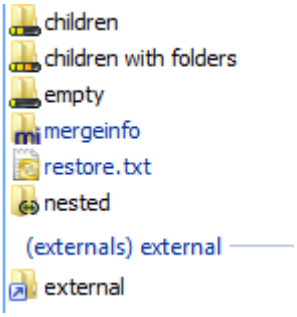
Zelena
: Elementi so spremenjeni krajevno in v skladišču. Ob posodobitvi bodo spremembe spojene. Situacija *lahko* povzroči spore pri posodobitvi.

Svetlo rdeča
: Elementi, ki so bili krajevno spremenjeni in izbrisani v skladišču ali spremenjeni v skladišču in izbrisani krajevno. Ob posodobitvi se spori *bodo* pojavili.

Črna
: Nespremenjeni elementi in elementi brez različic.

To je privzeta barvna shema, vendar lahko barve poljubno nastavite v oknu za nastavitve. Za več informacij preberite Razdelek 4.31.1.5, “Nastavitev barv TortoiseSVN”.

Overlay icons are used to indicate other states as well. The screenshot below shows all the possible overlays that are shown if necessary.



Overlays are shown for the following states:

- Checkout depth `empty`, meaning only the item itself.
- Checkout depth `files`, meaning only the item itself and all file children without child folders.
- Checkout depth `immediates`, meaning only the item itself and all file and folder children, but without children of the child folders.
- Nested items, i.e., working copies inside the working copy.
- External items, i.e., all items that are added via an `svn:externals` property.
- Items that are restored after a commit. See Razdelek 4.4.3, “Commit only parts of files” for details.
- Items that have property modifications, but only to the `svn:mergeinfo` property. If any other property is modified, the overlay is not used.

Items which have been switched to a different repository path are also indicated using an `(s)` marker. You may have switched something while working on a branch and forgotten to switch back to trunk. This is your warning sign! The context menu allows you to switch them back to the normal path again.

Iz kontekstnega menija lahko prikažete spremembe. Z uporabo Kontekstni meni → Primerjaj z osnovo preverite spremembe, ki ste jih naredili *vi*. Z ukazom Kontekstni meni → Pokaži razlike kot poenotene razlike preverite spremembe, ki so jih v skladišču naredili drugi uporabniki .

Posameznim datotekam lahko povrnete spremembe. Če ste pomotoma izbrisali datoteko, bo le-ta označena kot *manjkajoča*, zato lahko uporabite ukaz *Povrni*, da jo pridobite nazaj.

Datoteke brez različic in prezrte datoteke lahko pošljete v Koš z ukazom Kontekstni meni → Zbriši. Če želite datoteke zbrisati trajno (brez pošiljanja v Koš), ob kliku na Zbriši držite pritisnjeno tipko **Shift**.

If you want to examine a file in detail, you can drag it from here into another application such as a text editor or IDE, or you can save a copy simply by dragging it into a folder in explorer.

Stolpci so prilagodljivi. Če desno kliknete na glavo stolpca, se pojavi kontekstni meni, ki vam omogoča izbiro prikazanih stolpcev. Spremenite lahko tudi širino stolpcev z uporabo potega ročaja, ki se pojavi, ko greste z miško čez mejo stolpca. Spremembe se shranijo, tako da boste prilagojen pogled videli tudi naslednjič.

Če delate na več nepovezanih nalogah hkrati, lahko sestavljate datoteke skupaj v sezname sprememb. Za več informacij preberite Razdelek 4.4.2, “Seznami sprememb”.

Na dnu okna vidite povzetek območja revizij, ki jih uporabljate v delovni kopiji. To so *objavljene* revizije, ne *posodobljene* revizije; predstavljajo območje revizij, v katerih so bile datoteke nazadnje objavljene in ne revizije, pri katerih je bila narejena posodobitev delovne kopije. Območje revizij se nanaša le na prikazane elemente, ne na celovno delovno kopijo. Će želite, da se povzetek nanaša na celotno delovno kopijo, potrdite polje Prikaži nespremenjene datoteke.

Namig

Če želite videti ploski pogled delovne kopije (vse datoteke in mape na vseh nivojih hierarhije), to najlažje dosežete s pogovornim oknom Preveri spremembe. Edino, kar morate storiti, je, da potrdite polje Prikaži nespremenjene datoteke.

Popravljanje zunanjih preimenovanj

Včasih datoteke preimenujete zunaj sistema Subversion, zato se v seznamu sprememb pojavita manjkajoča datoteka in datoteka brez različic. Da bi preprečili izgubo zgodovine, morate sistemu Subversion povedati, da med datotekama obstaja povezava. Enostavno izberite tako staro ime (manjkajoče) kot novo ime (brez različic) in uporabite Kontekstni meni → Popravi premik in s tem povežete datoteki v preimenovanje.

Repairing External Copies

If you made a copy of a file but forgot to use the Subversion command to do so, you can repair that copy so the new file doesn't lose its history. Simply select both the old name (normal or modified) and the new name (unversioned) and use Context Menu → Repair Copy to pair the two files as a copy.

4.7.4. Pregledovanje razlik

Pogosto želite pogledati, kaj ste v datoteki spremenili. To storite z izbiro željene datoteke in uporabo ukaza Razlikuj iz kontekstnega menija. S tem poženete zunanji pregledovalnik razlik, ki bo primerjal trenutno datoteko z osnovno datoteko (revizija `BASE`), ki se je shranila, ko ste naredili zadnji prevzem ali posodobitev.

Namig

Tudi ko ne delate na delovni kopiji ali ko imate več verzij iste datoteke v mapi, lahko prikažete razlike:

V Raziskovalcu izberite datoteki, ki ju želite primerjati (n. pr. z uporabo tipke **Ctrl** in miške) in iz kontekstnega menija TortoiseSVN izberite Razlikuj. Datoteka, na katero ste kliknili nazadnje (datoteka, ki ima fokus oziroma datoteka s pikčasto obrobo) se bo obravnavala kot novejša.

4.8. Seznami sprememb

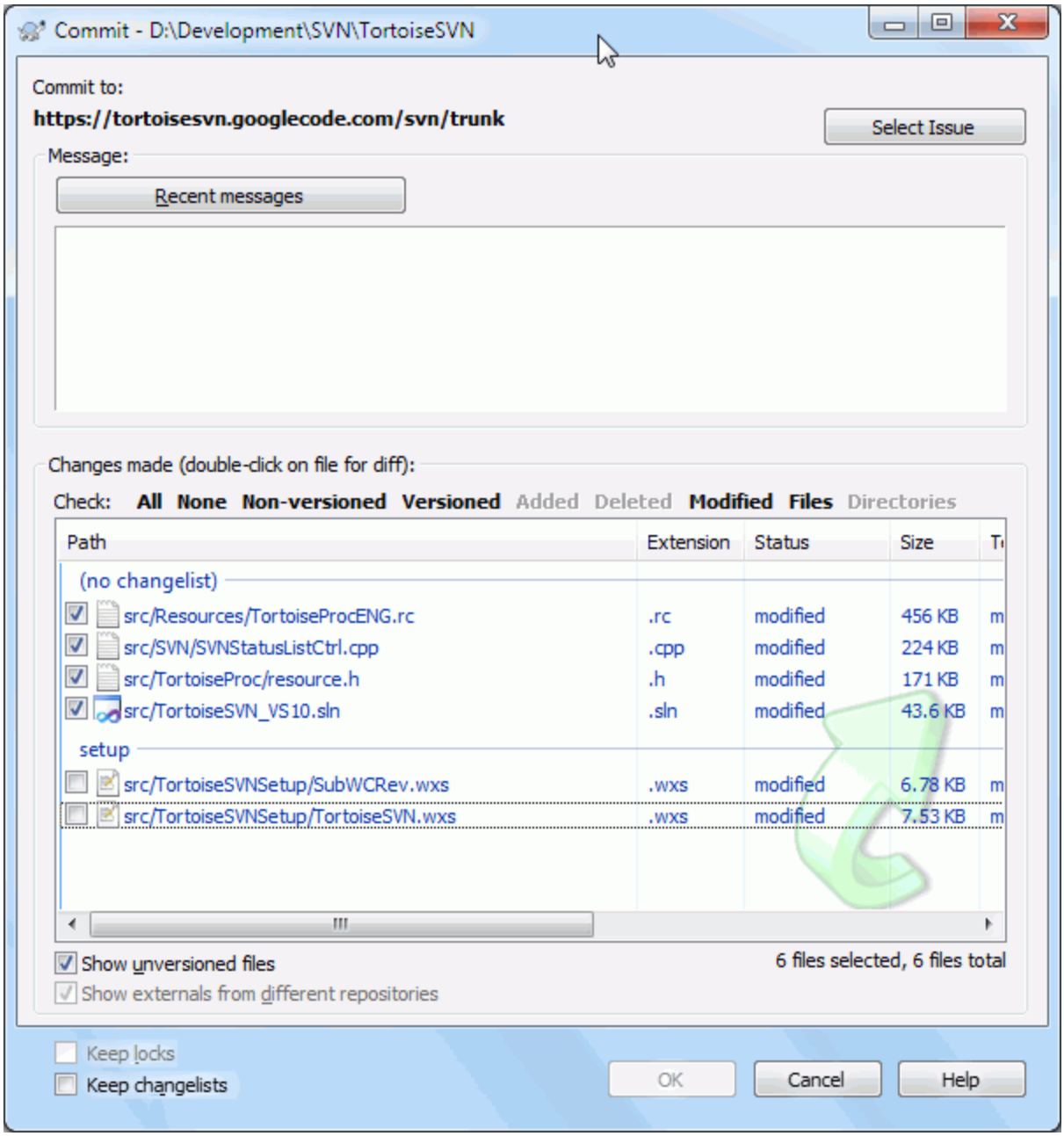
V idealnem svetu delate naenkrat na eni zadevi in vaša delovna kopija vsebuje le en skupek logičnih sprememb. V redu, nazaj v realnost. Večkrat se zgodi, da morate delati na več nepovezanih nalogah naenkrat in ko pogledate v okno za objave, vidite vse spremembe skupaj. Zmožnost *seznam sprememb* vam pomaga združiti datoteke v skupine. Tako lažje vidite, kaj delate. Seveda je to možno le, če se spremembe ne prekrivajo. Če dve različni nalogi zahtevata spremembo iste datoteke, sprememb ni mogoče ločiti.

Seznami sprememb se uporabljajo na več mestih, najbolj pa so pomembni pri preverjanju sprememb in pri objavi le-teh. Začnimo z oknom za preverjanje sprememb, ki po delu na več zmožnostih in več spremenjenih datotekah pokaže dolg seznam datotek. Sedaj želite spremenjene dodateke urediti in jih glede nato, katerim novostim pripadajo, postavtiti v ustrezne skupine.

Izberite eno ali več datotek in jih premaknite v seznam sprememb z uporabo ukaza Kontekstni meni → Premakni v seznam sprememb. Na začetku ni nobenega seznama sprememb, zato ga boste morali ob prvi uporabi ukaza ustvariti. Poimenujte ga smiselno glede na uporabo in kliknite gumb V redu. Okno za objave se spremeni in prikazuje skupine elementov.

Ko ste ustvarili seznam sprememb, lahko vanj dodate elemente. Vzamete jih lahko iz drugega seznama sprememb ali pa kar iz Raziskovalca. Dodajanje iz Raziskovalca je uporabno, saj lahko dodajate datoteke preden le-te sploh

spremenite. To lahko naredite tudi neposredno iz okna za preverjanje sprememb, vendar le, če imate vključeno možnost prikaza datotek brez različic.



Slika 4.15. Okno za objave s seznami sprememb

V pogovornem oknu za objave vidite iste datoteke, združene v sezname sprememb. Poleg vizualnega pregleda lahko skupine uporabljate tudi za hitro izbiro datotek za objavo s izbirom glave skupine.

TortoiseSVN si rezervira en seznam sprememb za lastno uporabo in sicer `ignore-on-commit`. Seznam se uporablja za datoteke, ki jih skoraj nikoli ne objavljate, čeprav imajo krajevne spremembe. Ta zmožnost je opisana v poglavju Razdelek 4.4.4, “Izključevanje elementov iz okna objav”.

Ob objavi datotek iz seznama sprememb običajno želite, da se te odstranijo iz seznama, tako da je to privzeta možnost pri vsaki objavi. Če želite, da datoteke ostanejo del seznama sprememb, potrdite polje Ohrani sezname sprememb na dnu pogovornega okna.

Namig

Seznami sprememb so povsem stvar odjemalca. Ustvarjanje in odstranjevanje seznamov sprememb ne vpliva na skladišče, prav tako pa tudi ne na delovne kopije drugih uporabnikov. Seznami sprememb so priročna možnost organiziranja spremenjenih datotek.

Pozor

Note that if you use changelists, externals will no longer show up in their own groups anymore. Once there are changelists, files and folders are grouped by changelist, not by external anymore.

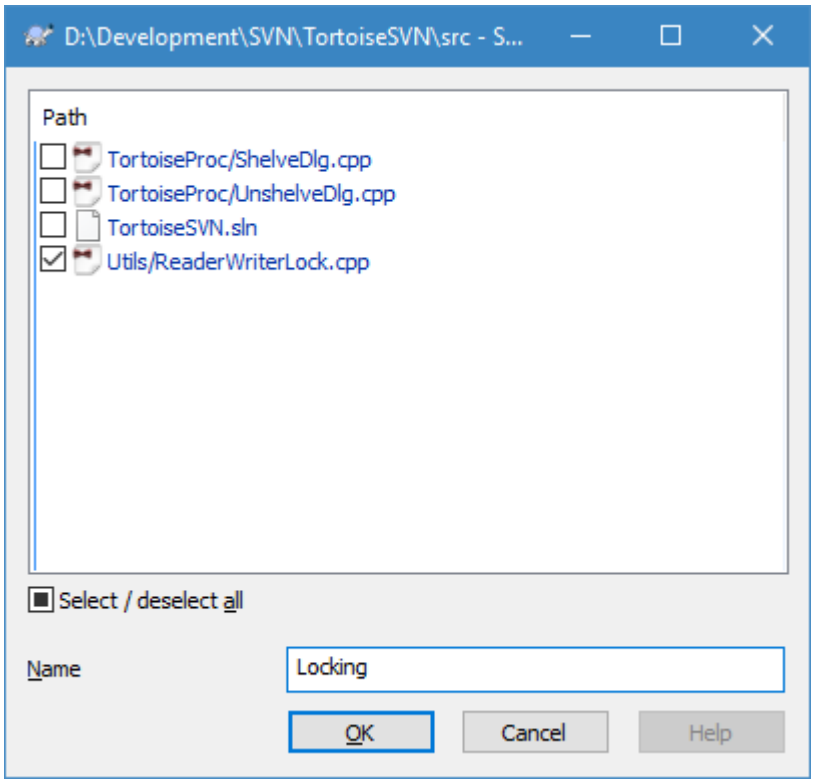
4.9. Shelving

More often than wanted, it's necessary to stop what you were working on and work on something else. For example a serious problem needs immediate dealing with and you have to stop working on the new feature. If possible, you should commit the changes you have done so far and then start working on the urgent issue, but often those changes would break the build or are just not ready for committing yet.

So if you can't commit your local changes yet, you have to put them aside while you're working on the urgent issue. The *shelving* feature helps you do exactly that: you can store your local changes on a shelve, get your working copy in a clean state again and work on the issue. After you're finished with the urgent issue and you've committed those changes, you can *unshelve* your shelved work and continue working on your previous task again.

Two new commands are implemented for this. One for shelving and one for unshelving.

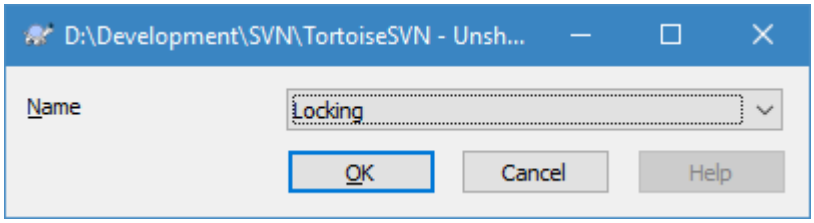
To shelve your local changes, select your working copy and use Context Menu → Shelve The following dialog allows you to select the files you want to shelve and give a name under which you want to store them.



Slika 4.16. Shelve dialog

Then click OK and the changes are stored under the name you specified, and your working copy is reset to a clean state.

To unshelve your changes, use Context Menu → Unshelve to get the unshelve dialog. This dialog shows you a list of all shelved items. Select the shelved item you want to apply back to your working copy and click OK.



Slika 4.17. Unshelve dialog

Namig

Shelves are purely a local client feature. Creating and removing Shelves will not affect the repository, nor anyone else's working copy.

Pozor

Currently only text files can be shelved and unshelved. Binary files and text files encoded in utf-16 (unicode) can not be shelved.

4.10. Pogovorno okno Dnevnik

Za vsako objavljeno spremembo vnesite sporočilo dnevniškega zapisa. Tako lahko kasneje ugotovite, katere spremembe ste kdaj naredili in imate podroben pregled celotnega razvojnega procesa.

Pogovorno okno dnevnika pridobi dnevniške zapise in jih prikaže. Zaslon je razdeljen na tri dele.

- Zgornji del prikazuje seznam revizij, kjer je bila datoteka/mapa objavljena. Povzetek vsebuje čas in datum, avtorja objavljene spremembe in začetni del sporočila dnevnika.

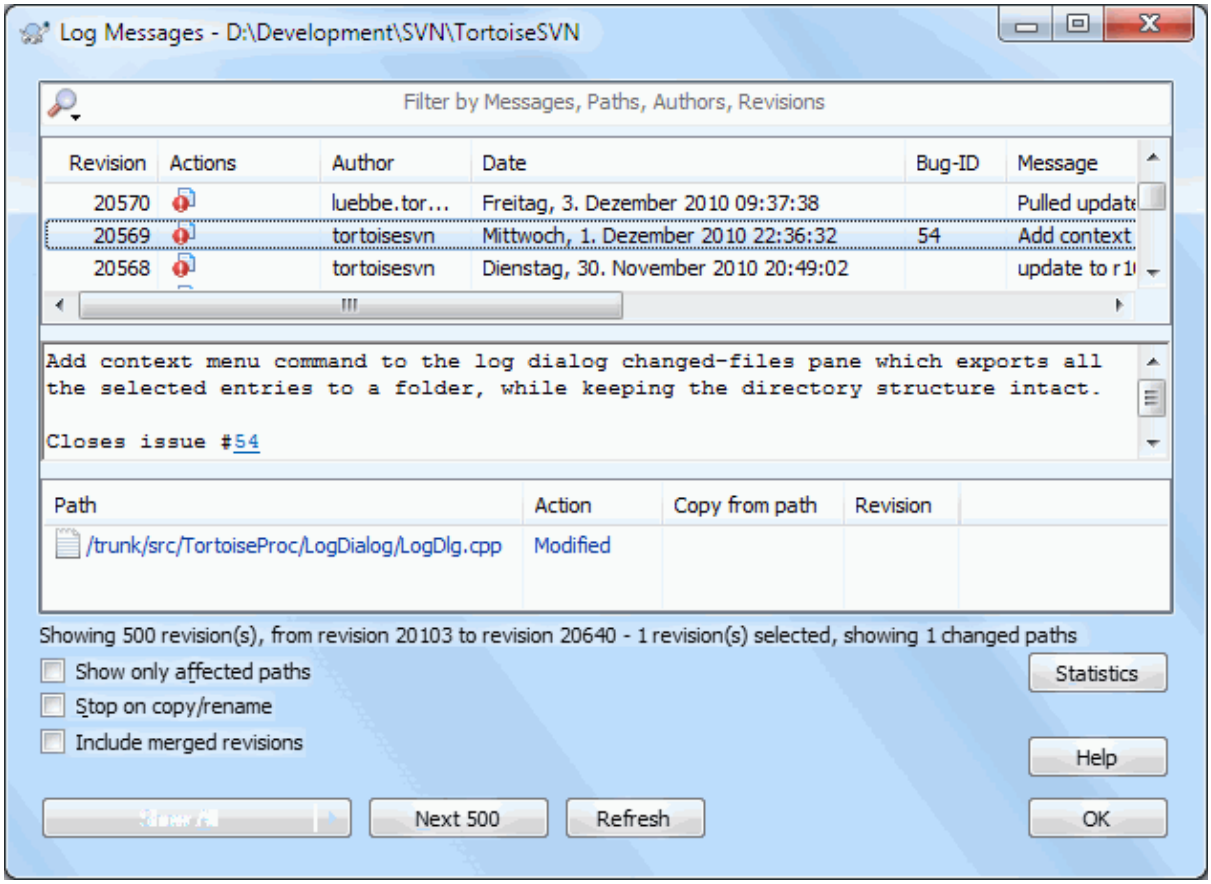
  Vrstice, obarvane z modro barvo, pomenijo, da je bil nek element prekopiran na to vejo razvoja (mogoče iz druge veje).

- Srednji del prikazuje sporočilo dnevniškega zapisa za izbrano revizijo.

- Spodnji del prikazuje seznam datotek in map, ki so bile spremenjene v izbrani reviziji.

Pogovorno okno pa naredi še veliko več - ponudi vam kontekstni meni, ki prinaša še več informacij o zgodovini projekta.

4.10.1. Klicanje pogovornega okna dnevniških zapisov



Slika 4.18. Okno dnevnika

Obstaja veliko mest, od koder lahko lahko pokličete okno dnevnika:

- Iz kontekstnega podmenija TortoiseSVN
- Iz strani lastnosti
- Iz pogovornega okna napredka po posodobitvi delovne kopije. V tem primeru Dnevnik prikaže le tiste revizije, ki so se spremenile od zadnje posodobitve.
- From the repository browser

If the repository is unavailable you will see the Want to go offline? dialog, described in Razdelek 4.10.10, “Nepovezan način”.

4.10.2. Akcije dnevnika

Zgornji del okna vsebuje stolpec Dejanja z ikonami, ki povzemajo, kaj se je v določeni reviziji zgodilo. Obstajajo štiri različne ikone, vsaka je prikazana v svojem stopcu.

Če je v reviziji prišlo do spremembe datoteke ali mape, se v prvem stolpcu prikaže ikona *spremenjeno*.

Če je bila v reviziji dodana datoteka ali mapa, se v drugem stolpcu prikaže ikona `dodano`.

Če je bila v reviziji izbrisana datoteka ali mapa, se v tretjem stolpcu prikaže ikona `zbrisano`.

Če je revizija zamenjala datoteko ali mapo, se v četrtem stolcu prikaže ikona `zamenjano`.

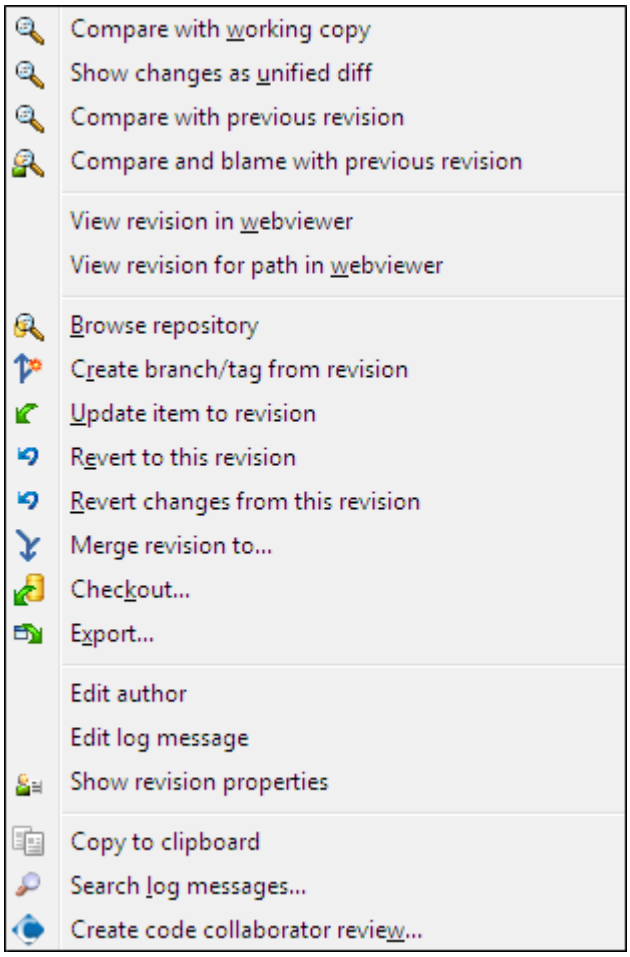
If a revision moved or renamed a file or directory, the *moved* icon is shown in the fourth column.

If a revision replaced a file or directory by moving/renaming it, the *move replaced* icon is shown in the fourth column.

If a revision merged a file or directory, the *merged* icon is shown in the fourth column.

If a revision reverse merged a file or directory, the *reverse merged* icon is shown in the fourth column.

4.10.3. Pridobivanje dodatnih informacij



Slika 4.19. Zgornji del Dnevnika s kontekstnim menijem

The top pane of the Log dialog has a context menu that allows you to access much more information. Some of these menu entries appear only when the log is shown for a file, and some only when the log is shown for a folder.

Compare with working copy
: Primerjate izbrane revizije z delovno kopijo. Privzeto orodje za razlikovanje je TortoiseMerge, ki je del paketa TortoiseSVN. Če ste zahtevali dnevnik za mapo, se bo prikazal seznam spremenjenih datotek, vi pa boste lahko pregledali narejene spremembe za vsako datoteko posebej.

Compare and blame with working BASE
: Blame the selected revision, and the file in your working BASE and compare the blame reports using a visual diff tool. Read Razdelek 4.24.2, “Okrivi spremembe” for more detail. (files only)

Show changes as unified diff
: Pogledate spremembe v izbranih revizijah v obliki poenotene različice (oblika popravkov GNU). Prikaže le razlike in nekaj bližnjih vrstic. Ta oblika je težje berljiva kot vizualna primerjava, vendar prikaže vse spremembe v datoteki v zgoščeni obliki.

  If you hold down the **Shift** key when clicking on the menu item, a dialog shows up first where you can set options for the unified diff. These options include the ability to ignore changes in line endings and whitespaces.

Primerjaj s predhodno revizijo
: Compare the selected revision with the previous revision. This works in a similar manner to comparing with your working copy. For folders this option will first show the changed files dialog allowing you to select files to compare.

Primerjaj in okrivi s predhodno revizijo
: Show the changed files dialog allowing you to select files. Blame the selected revision, and the previous revision, and compare the results using a visual diff tool. (folders only)

Save revision to...
: Save the selected revision to a file so you have an older version of that file. (files only)

Open / Open with...
: Open the selected file, either with the default viewer for that file type, or with a program you choose. (files only)

Okrivi...
: Blame the file up to the selected revision. (files only)

Browse repository
: Odprete brskalnik po skladišču za ogled izbrane datoteke ali mape, kakršna je bila ob izbrani reviziji.

Create branch/tag from revision
: Create a branch or tag from a selected revision. This is useful e.g. if you forgot to create a tag and already committed some changes which weren't supposed to get into that release.

Update item to revision
: Update your working copy to the selected revision. Useful if you want to have your working copy reflect a time in the past, or if there have been further commits to the repository and you want to update your working copy one step at a time. It is best to update a whole directory in your working copy, not just one file, otherwise your working copy could be inconsistent.

  If you want to undo an earlier change permanently, use Revert to this revision instead.

Revert to this revision
: Revert to an earlier revision. If you have made several changes, and then decide that you really want to go back to how things were in revision N, this is the command you need. The changes are undone in your working copy so this operation does *not* affect the repository until you commit the changes. Note that this will undo *all* changes made after the selected revision, replacing the file/folder with the earlier version.

  If your working copy is in an unmodified state, after you perform this action your working copy will show as modified. If you already have local changes, this command will merge the *undo* changes into your working copy.

What is happening internally is that Subversion performs a reverse merge of all the changes made after the selected revision, undoing the effect of those previous commits.

If after performing this action you decide that you want to *undo the undo* and get your working copy back to its previous unmodified state, you should use TortoiseSVN → Revert from within Windows Explorer, which will discard the local modifications made by this reverse merge action.

If you simply want to see what a file or folder looked like at an earlier revision, use Update to revision or Save revision as... instead.

Revert changes from this revision
: Undo changes from which were made in the selected revision. The changes are undone in your working copy so this operation does *not* affect the repository at all! Note that this will undo the changes made in that revision only; it does not replace your working copy with the entire file at the earlier revision. This is very useful for undoing an earlier change when other unrelated changes have been made since.

  If your working copy is in an unmodified state, after you perform this action your working copy will show as modified. If you already have local changes, this command will merge the *undo* changes into your working copy.

  What is happening internally is that Subversion performs a reverse merge of that one revision, undoing its effect from a previous commit.

  You can *undo the undo* as described above in Revert to this revision.

Spoji revizijo v...
: Merge the selected revision(s) into a different working copy. A folder selection dialog allows you to choose the working copy to merge into, but after that there is no confirmation dialog, nor any opportunity to try a test merge. It is a good idea to merge into an unmodified working copy so that you can revert the changes if it doesn't work out! This is a useful feature if you want to merge selected revisions from one branch to another.

Checkout...
: Naredite svež prevzem izbrane mape in izbrane revizije. S tem se prikaže okno, v katerem potrdite naslov URL in revizijo in izberete lokacijo za prevzem.

Export...
: Izvozite izbrane datoteke/mape v izbrani reviziji. Pokaže se pogovorno okno z naslovom URL in revizijo, kjer vnesete še lokacijo izvoza.

Edit author / log message
: Uredite sporočilo dnevniškega zapisa in avtoja v predhodni objavi. Za dodatne informacije preberite Razdelek 4.10.7, “Spreminjanje sporočila dnevniškega zapisa in avtorja”.

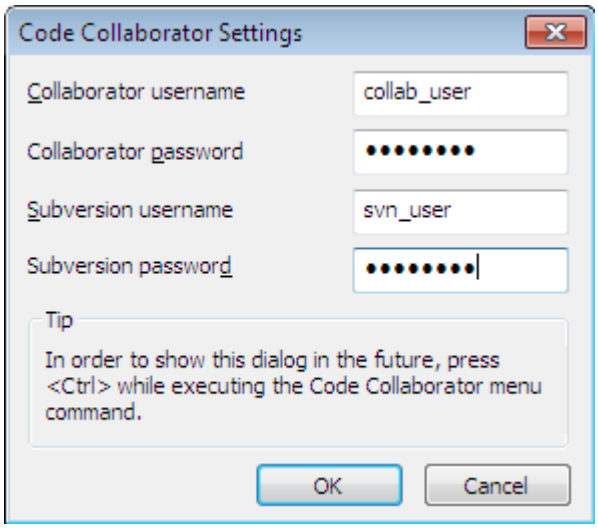
Pokaži lastnosti revizije
: View and edit any revision property, not just log message and author. Refer to Razdelek 4.10.7, “Spreminjanje sporočila dnevniškega zapisa in avtorja”.

Kopiraj na odložišče
: Prekopirate podrobnosti dnevnika izbranih revizij na odložišče. S tem prekopirate številko revizije, avtorja, datum, sporočilo in seznam spremenjenih elementov za vsako revizijo.

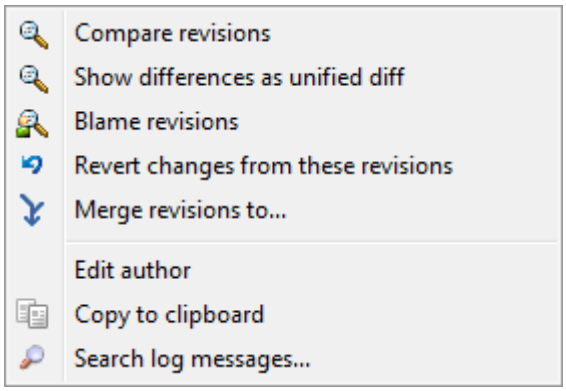
Search log messages...
: Iščete po sporočilih dnevniških zapisov. Iskanje poteka po dnevniških zapisih, ki jih vnesete, in po povzetkih akcij, ki jih ustvari Subversion in so prikazani v spodnjem delu okna. Iskanje ni občutljivo na velike in male črke.

Create code collaborator review...
: This menu is shown only if the SmartBear code collaborator tool is installed. When invoked for the first time, a dialog is shown prompting the user to enter user credentials for both code collaborator and SVN. Once the

settings are stored, the settings dialog is no longer shown when the menu is invoked, unless the user holds **Ctrl** while executing the menu item. The configuration and the chosen revision(s) are used to invoke the code collaborator graphical user interface client, which creates a new review with the selected revisions.



Slika 4.20. The Code Collaborator Settings Dialog



Slika 4.21. Kontekstni meni v zgornjem delu okna v primeru dveh izbranih revizij

Če izberete dve reviziji naenkrat (z uporabo tipke **Ctrl**), se kontekstni meni spremeni in ponudi manj možnosti. Lahko:

Compare revisions
: Primerjate izbrani reviziji z grafičnim orodjem za razlikovanje. Privzeto orodje je TortoiseMerge, ki se namesti hkrati z orodjem TortoiseSVN.

  Če izberete to možnost za mapo, se pojavi novo okno, ki vsebuje seznam datotek, za katere lahko pogledate spremembe. Več o primerjanju revizij si lahko preberete v Razdelek 4.11.3, “Primerjanje map”.

Blame revisions
: Okrivite dve reviziji in primerjate rezultate z grafičnim orodjem za razlikovanje. Za več informacij preberite Razdelek 4.24.2, “Okrivi spremembe”.

Show differences as unified diff
: Pogledate razlike med izbranima revizijama v obliki poenotene razlike. Deluje za datoteke in mape.

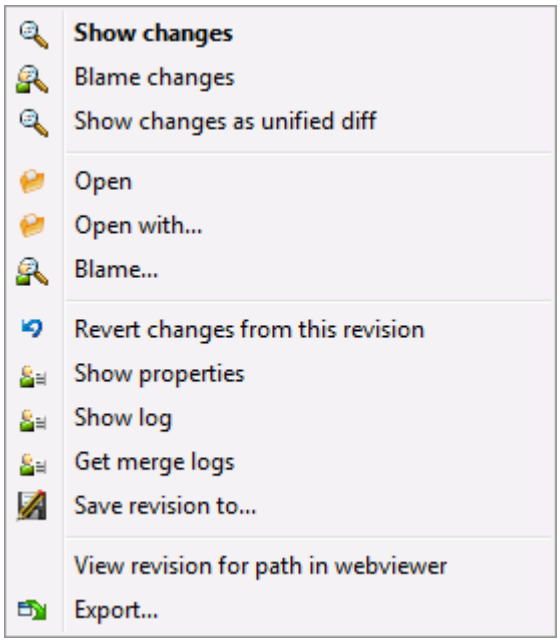
Kopiraj na odložišče
: Skopirate sporočila dnevniških zapisov na odložišče, kot je to opisano zgoraj.

Search log messages...
: Iščete med sporočili dnevniških zapisov, kot je opisano zgoraj.

Če izberete več revizij (z uporabo tipk **Ctrl** ali **Shift**), se v kontekstnem meniju pojavi možnost, ki vam omogoča povrnitev vseh sprememb, ki so bile narejene v teh revizijah. To je najlažji način, kako povrniti spremembe iz večih revizij v eni potezi.

Izberete pa lahko tudi spajanje izbranih revizij v drugo delovno kopijo, kot je opisano zgoraj.

If all selected revisions have the same author, you can edit the author of all those revisions in one go.



Slika 4.22. Kontekstni meni spodnjega dela Dnevnika

The bottom pane of the Log dialog also has a context menu that allows you to

Show changes
: Show changes made in the selected revision for the selected file.

Blame changes
: Okrivite izbrano revizijo in predhodnjo revizijo izbrane datoteke in prikažete razlike z grafičnim orodjem za razlikovanje. Za več informacij preberite Razdelek 4.24.2, “Okrivi spremembe”.

Show as unified diff
: Show file changes in unified diff format. This context menu is only available for files shown as *modified*.

Open / Open with...
: Odprete izbrano datoteko s privzetim pregledovalnikom ali programom po lastni izbiri.

Okrivi...
: Opens the Blame dialog, allowing you to blame up to the selected revision.

Revert changes from this revision
: Povrnte spremembe, narejene na izbrani datoteki izbrane revizije.

Show properties
: Pogledate lastnosti Subversion za izbran element.

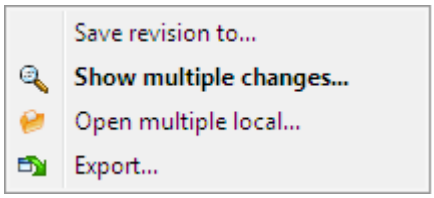
Show log
: Prikažete dnevnik za izbrano datoteko.

Priskrbi dnevnike spajanj
: Show the revision log for the selected single file, including merged changes. Find out more in Razdelek 4.10.6, “Zmožnosti sledenja spajanja”.

Save revision to...
: Shranite izbrano revizijo datoteke, tako da imate starejšo različico te datoteke.

Export...
: Export the selected items in this revision to a folder, preserving the file hierarchy.

When multiple files are selected in the bottom pane of the Log dialog, the context menu changes to the following:



Slika 4.23. The Log Dialog Bottom Pane with Context Menu When Multiple Files Selected.

Save revision to...
: Shranite izbrano revizijo datoteke, tako da imate starejšo različico te datoteke.

Show multiple changes...
: Show changes made in the selected revision for the selected files. Note that the show changes functionality is invoked multiple times, which may bring up multiple copies of your selected diff tool, or just add a new comparison tab in your diff tool. If you have selected more than 15 files, you will be prompted to confirm the action.

Open multiple local...
: This will open local working copy files that correspond to your selected files using the application that is registered for the extension. [The behavior is the one you would get double-clicking the working-copy file(s) in Windows explorer]. Depending on how your file extension is associated to an application and the capabilities of the application, this may be a slow operation. In the worst case, new instances of the application may be launched by Windows for each file that was selected.

  If you hold **Ctrl** while invoking this command, the working copy files are always loaded into Visual Studio. This only works when the following conditions are met: Visual Studio must be running in the same user context while having the same process integrity level [running as admin or not] as TortoiseProc.exe. It may be desirable to have the solution containing the changed files loaded, although this is not strictly necessary. Only files that exist on disk with extensions [.cpp, .h, .cs, .rc, .resx, .xaml, .js, .html, .htm, .asp, .aspx, .php, .css and .xml] will be loaded. A maximum of 100 files can be loaded into Visual Studio at one time, and the files are always loaded as new tabs into the currently open instance of Visual Studio. The benefit of reviewing code changes in Visual Studio lies in the fact that you can then use the built-in code navigation, reference finding, static code analysis and other tools built into Visual Studio.

Export...
: Export the selected files/folder at the selected revision. This brings up a dialog for you to confirm the URL and revision, and select a location for the export.

Namig

Mogoče ste opazili, da uporabljamo izraza razlike in spremembe. Kakšna je razlika?

Subversion uses revision numbers to mean 2 different things. A revision generally represents the state of the repository at a point in time, but it can also be used to represent the changeset which

created that revision, e.g. “Done in r1234” means that the changes committed in r1234 implement feature X. To make it clearer which sense is being used, we use two different terms.

Če izberete revizijo N in M, vam kontekstni meni ponuja možnost prikaza *razlik* med tema dvema revizijama. Ekvivalenten ukaz pri delu z odjemalcem za ukazno vrstivo je `diff -r M:N`.

Če izberete samo revizijo N, vam kontekstni meni ponuja prikaz *sprememb*, opravljenih v tej reviziji. Ekvivalentna ukaza pri uporabi odjemalca za ukazno vrstico sta `diff -r N-1:N` in `diff -c N`.

Spodnja polovica prikazuje vse spremenjene datoteke v vseh izbranih revizijah. Kontekstni meni ponuja prikaz *sprememb*.

4.10.4. Pridobivanje dodatnih dnevniških zapisov

Okno Dnevnik ne pokaže vedno vseh sprememb. Razlogi so naslednji:

- V večjih skladiščih se lahko nahaja stotine ali tisoče sprememb. Pridobivanje informacij o vseh spremembah lahko traja dolgo časa. Običajno pa vas zanimajo le zadnje spremembe. Privzeta nastavitev števila dnevniških zapisov za prikaz je 100, vendar lahko to vrednost spremenite v TortoiseSVN → Nastavitve (Razdelek 4.31.1.2, “Pogovorna okna 1”),
- Če je potrjeno polje Ustavi ob kopiranju/preimenovanju, Dnevnik prikaže le revizije do trenutka, ko je bila izbrana datoteka ali mapa prekopirana iz neke druge lokacije v skladišču. To je uporabno, kadar pregledujete veje (ali oznake), saj se prikažejo le spremembe, ki so bile narejene na tej veji.

  Običajno boste to možnost pustili izklopljeno. TortoiseSVN si zapomni stanje potrditvenih polj, tako da bo upošteval vaše nastavitve.

  Kadar je okno Prikaži dnevnik prikazano iz okna za spajanje, je to polje vedno potrjeno. Vzrok je v tem, da spajanje večinoma deluje na spremembah na vejah, zato uporaba map pred korensko mapo v tem primeru nima nobenega smisla.

  Upoštevajte, da Subversion izvaja preimenovanje s parom kopirane in izbrisane datoteke, zato bo preimenovanje datoteke ali mape povzročilo, da se v tem primeru dnevniški zapis ustavi, če je ta možnost vklopljena.

Če želite videti več dnevniških zapisov, kliknite na gumb Naslednjih 100 in prikazalo se bo naslednjih 100 zapisov. Postopek lahko ponovite poljubno mnogokrat.

Zraven tega gumba se nahaja večfunkcijski gumb, ki si zapomni zadnjo akcijo, za katero je bil uporabljen. Kliknite na puščico, če želite videti še ostale akcije, ki so vam na razpolago.

Uporabite gumb Pokaži obseg..., če želite videti le določen obseg revizij. V okno boste vnesli začetno in končno revizijo.

Uporabite gumb Prikaži vse, če želite videti *vse* dnevniške zapise od končne revizije HEAD do revizije 1.

To refresh the latest revision in case there were other commits while the log dialog was open, hit the **F5** key.

To refresh the log cache, hit the **Ctrl-F5** keys.

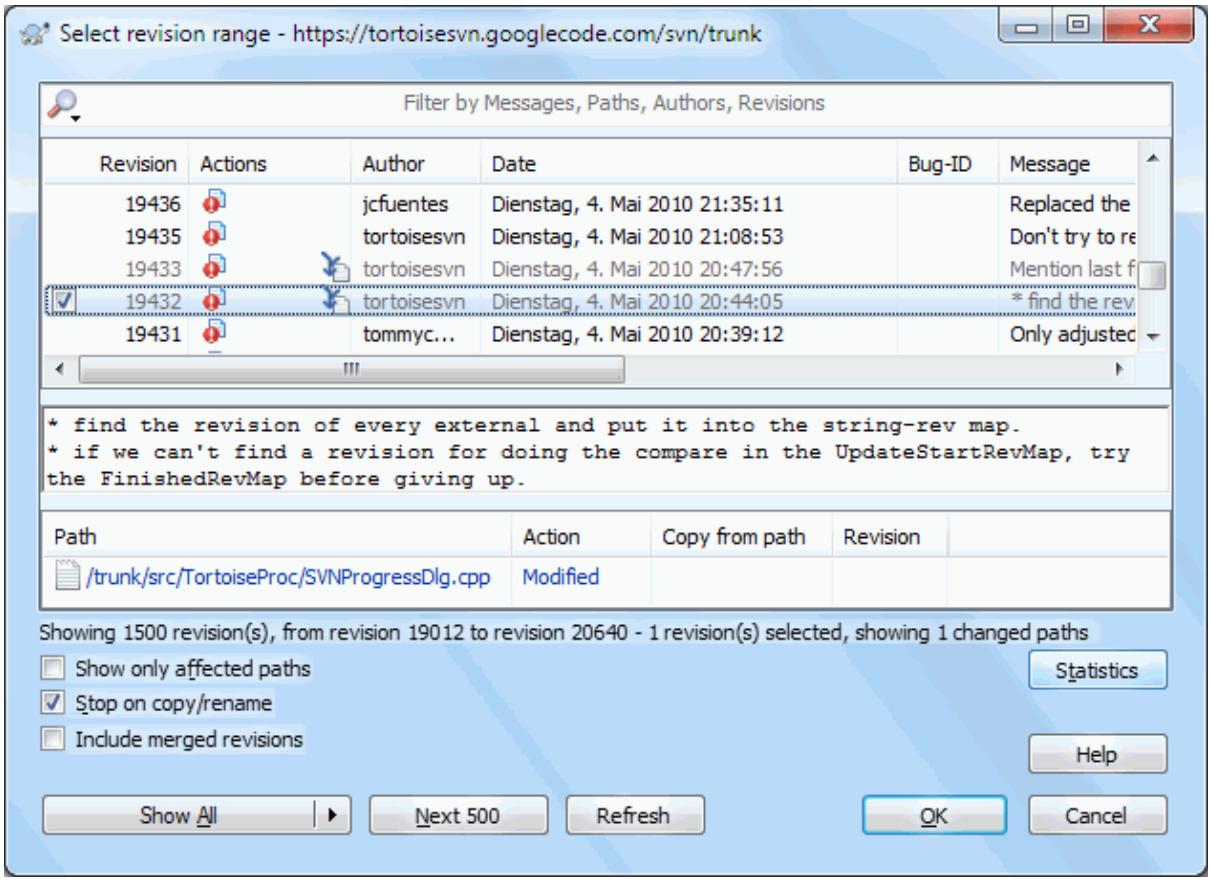
4.10.5. Current Working Copy Revision

Because the log dialog shows you the log from HEAD, not from the current working copy revision, it often happens that there are log messages shown for content which has not yet been updated in your working copy. To help make this clearer, the commit message which corresponds to the revision you have in your working copy is shown in bold.

When you show the log for a folder the revision highlighted is the highest revision found anywhere within that folder, which requires a crawl of the working copy. The crawl takes place within a separate thread so as not to delay showing the log, but as a result highlighting for folders may not appear immediately.

4.10.6. Zmožnosti sledenja spajanja

Subversion 1.5 uvaja sledenje spajanja z uporabo lastnosti. To nam omogoča pridobiti podrobnejšo zgodovino spojenih sprememb. Primer: če na posebni veji razvijate neko novo zmožnost programa in potem spremembe spojite na glavno vejo, boste na glavni veji to novo zmožnost videli kot eno objavo, čeprav ste pri razvoju te lastnosti na posebni veji naredili tisoč objav.



Slika 4.24. Dnevnik prikazuje sledenje spajanja revizij

Če želite videti, katere revizije so bile spojene kot del objave, uporabite potrditveno polje Vključi spojene revizije. Dnevniška sporočila se bodo ponovno prenesla, tokrat bodo vključeni tudi zapisi revizij, iz katerih je bilo narejeno spajanje. Spojene revizije so prikazane v sivi barvi, saj predstavljajo spremembe, narejene na drugi veji.

Spajanje seveda nikoli ni enostavno! Med razvojem na veji boste na vejo občasno spajali spremembe iz glavne veje. Torej bo zgodovina spajanja glavne veje vsebovala dodaten nivo zgodovine spajanja. Različni nivoji spajanja so v dnevniku prikazani z zamikanjem.

4.10.7. Spreminjanje sporočila dnevniškega zapisa in avtorja

Revision properties are completely different from the Subversion properties of each item. Revprops are descriptive items which are associated with one specific revision number in the repository, such as log message, commit date and committer name (author).

Včasih želite spremeniti vnešeno sporočilo dnevniškega zapisa - zaradi napake v črkovanju, ker boste želeli izboljšati komentar ali zaradi kakega drugega razloga. Ali pa boste želeli spremeniti avtorja objave, ker ste si nastavili napačno avtentikacijo ali pa...

Subversion lets you change revision properties any time you want. But since such changes can't be undone (those changes are not versioned) this feature is disabled by default. To make this work, you must set up a pre-revprop-change hook. Please refer to the chapter on *Hook Scripts* [http://svnbook.red-bean.com/en/1.8/svn.reposadmin.create.html#svn.reposadmin.create.hooks] in the Subversion Book for details about how to do that. Read Razdelek 3.3, “Server side hook scripts” to find some further notes on implementing hooks on a Windows machine.

Once you've set up your server with the required hooks, you can change the author and log message (or any other revprop) of any revision, using the context menu from the top pane of the Log dialog. You can also edit a log message using the context menu for the middle pane.

Pozor

Lastnosti revizij v sistemu Subversion so brez različic, zato bo sprememba lastnosti (n. pr. lastnost `svn:log` za sporočilo objave) *za vedno* prepisala predhodno vrednosti te lastnosti.

Pomembno

Since TortoiseSVN keeps a cache of all the log information, edits made for author and log messages will only show up on your local installation. Other users using TortoiseSVN will still see the cached (old) authors and log messages until they refresh the log cache. Refer to Razdelek 4.10.11, “Osveževanje pogleda”

4.10.8. Filtriranje dnevniških zapisov

Če želite omejiti izpise na le tiste, ki vas zanimajo, in se tako izogniti brskanju po seznamu, lahko uporabite možnosti filtriranja na vrhu okna. Začetni in končni datum vam omogočata omejitev vpisov na omejeno časovno območje, polje za iskanje pa vam omogoča izpis le tistih revizij, ki vsebujejo iskani niz.

Click on the search icon to select which information you want to search in, and to choose *regex* mode. Normally you will only need a simple sub-string search, but if you need to more flexible search terms, you can use regular expressions. If you hover the mouse over the box, a tooltip will give hints on how to use the regex functions, or the sub-string functions. The filter works by checking whether your filter string matches the log entries, and then only those entries which *match* the filter string are shown.

Simple sub-string search works in a manner similar to a search engine. Strings to search for are separated by spaces, and all strings must match. You can use a leading - to specify that a particular sub-string is not found (invert matching for that term), and you can use `!` at the start of the expression to invert matching for the entire expression. You can use a leading + to specify that a sub-string should be included, even if previously excluded with a -. Note that the order of inclusion/exclusion is significant here. You can use quote marks to surround a string which must contain spaces, and if you want to search for a literal quotation mark you can use two quotation marks together as a self-escaping sequence. Note that the backslash character is *not* used as an escape character and has no special significance in simple sub-string searches. Examples will make this easier:

```
Alice Bob -Eve
```

searches for strings containing both Alice and Bob but not Eve

```
Alice -Bob +Eve
```

searches for strings containing both Alice but not Bob, or strings which contain Eve.

```
-Case +SpecialCase
```

searches for strings which do not contain Case, but still include strings which contain SpecialCase.

```
!Alice Bob
```

searches for strings which do not contain both Alice and Bob

```
!-Alice -Bob
```

do you remember De Morgan's theorem? NOT(NOT Alice AND NOT Bob) reduces to (Alice OR Bob).

```
"Alice and Bob"
```

searches for the literal expression “Alice and Bob”

```
""
```

searches for a double-quote anywhere in the text

```
"Alice says ""hi"" to Bob"
```

searches for the literal expression “Alice says "hi" to Bob”.

Describing the use of regular expression searches is beyond the scope of this manual, but you can find online documentation and a tutorial at http://www.regular-expressions.info/.

Upoštevajte, da filter deluje na sporočilih, ki so že na razpolago. Filter ne nadzira prenosa sporočil iz skladišča.

You can also filter the path names in the bottom pane using the Show only affected paths checkbox. Affected paths are those which contain the path used to display the log. If you fetch the log for a folder, that means anything in that folder or below it. For a file it means just that one file. Normally the path list shows any other paths which are affected by the same commit, but in grey. If the box is checked, those paths are hidden.

Včasih delo na projektu zahteva uporabo predpisane oblike sporočila dnevniškega zapisa, kar pomeni, da besedilo, ki opisuje spremembe, ni vidno pri skrajšanem povzetku, ki se prikaže v zgornjem delu okna. Lastnost `tsvn:logsummary` lahko uporabite, da izvlečete del besedila, ki se prikaže kot povzetek. Za več informacij o tej lastnosti reberite Razdelek 4.18.2, “Projektne lastnosti TortoiseSVN”.

No Log Formatting from Repository Browser

Because the formatting depends upon accessing Subversion properties, you will only see the results when using a checked out working copy. Fetching properties remotely is a slow operation, so you will not see this feature in action from the repo browser.

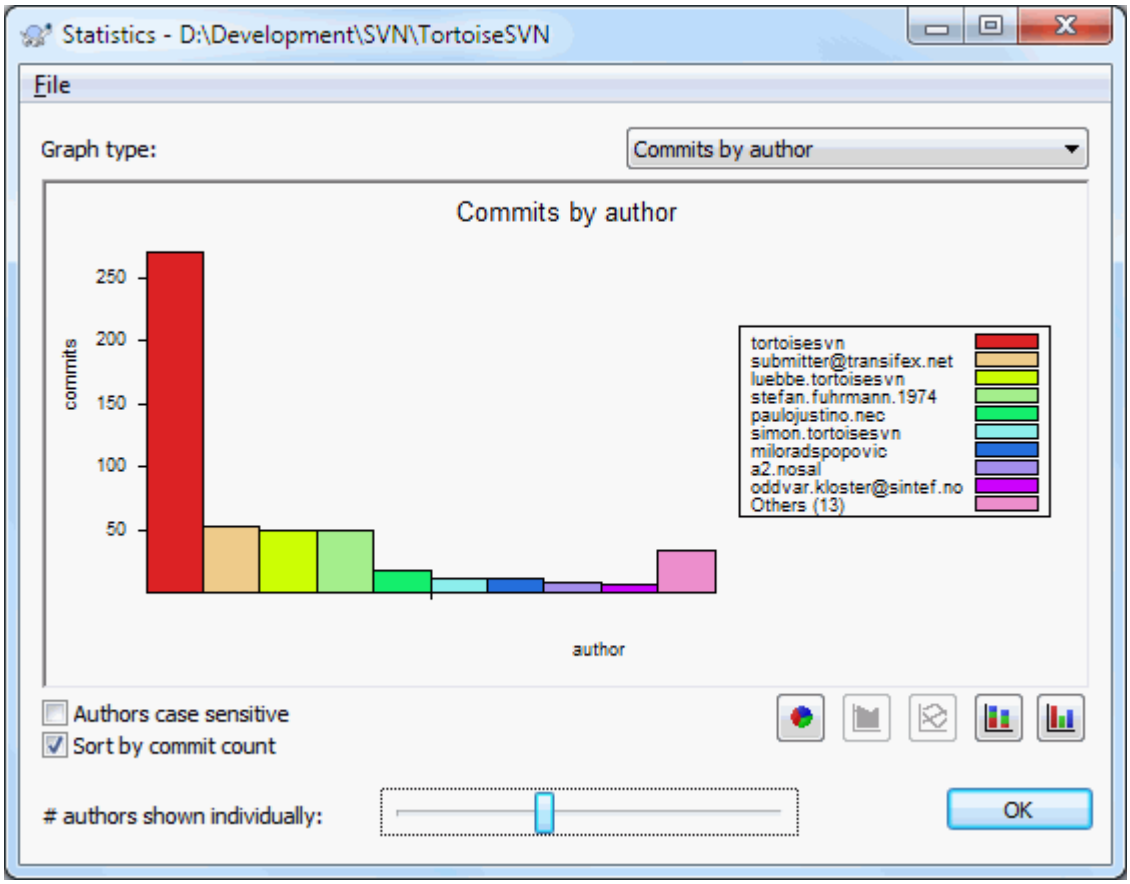
4.10.9. Statistične informacije

Klik na gumb Statistika postreže z nekaj zanimivimi statističnimi informacijami o revizijah, prikazanih v oknu dnevnika. Prikazano je, koliko avtorjev je sodelovalo, koliko objav so naredili, napredek po tednih in še veliko več. Sedaj lahko enostavno vidite, kdo je delal največ in kdo je zabušaval;)

4.10.9.1. Statistika

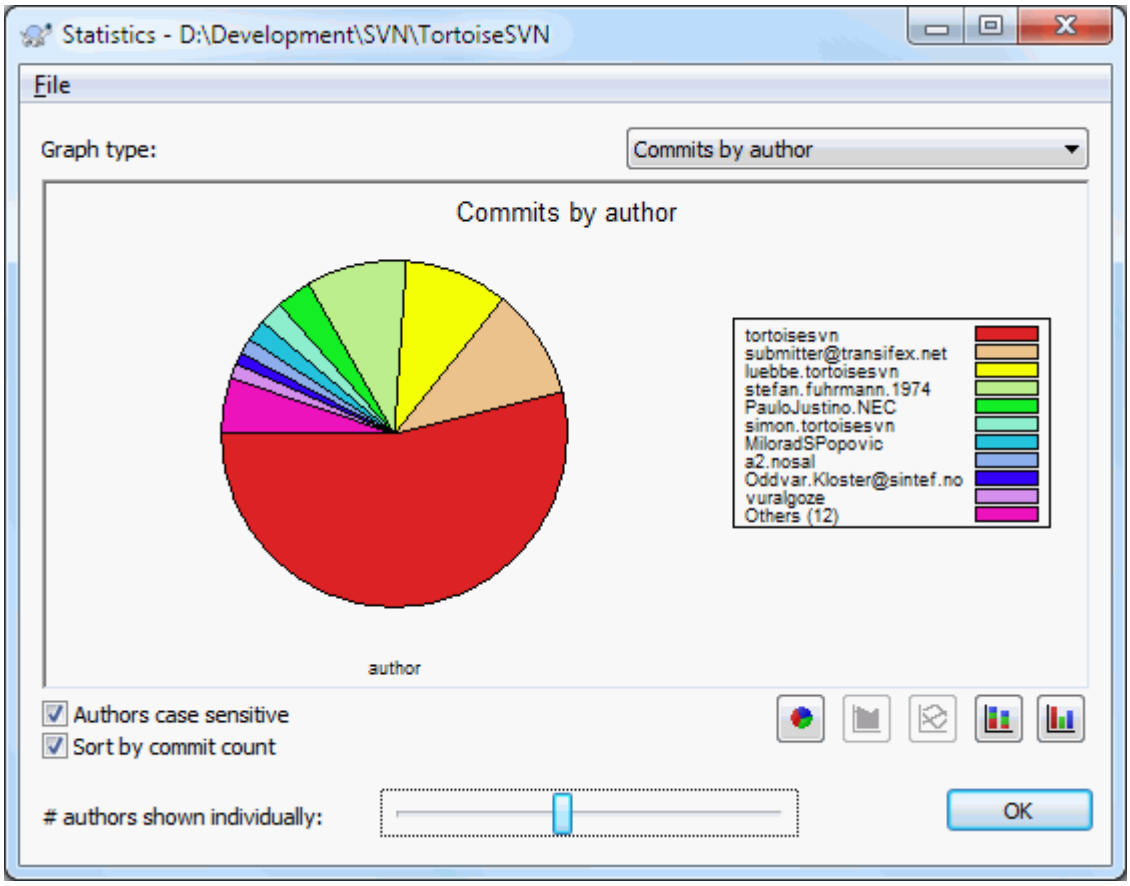
Ta stran vam pokaže vse mogoče številke, predvsem časovno območje in število revizij ter nekatere ekstremne in povprečne vrednosti.

4.10.9.2. Stran objav glede na avtorja



Slika 4.25. Histogram objav glede na avtorja

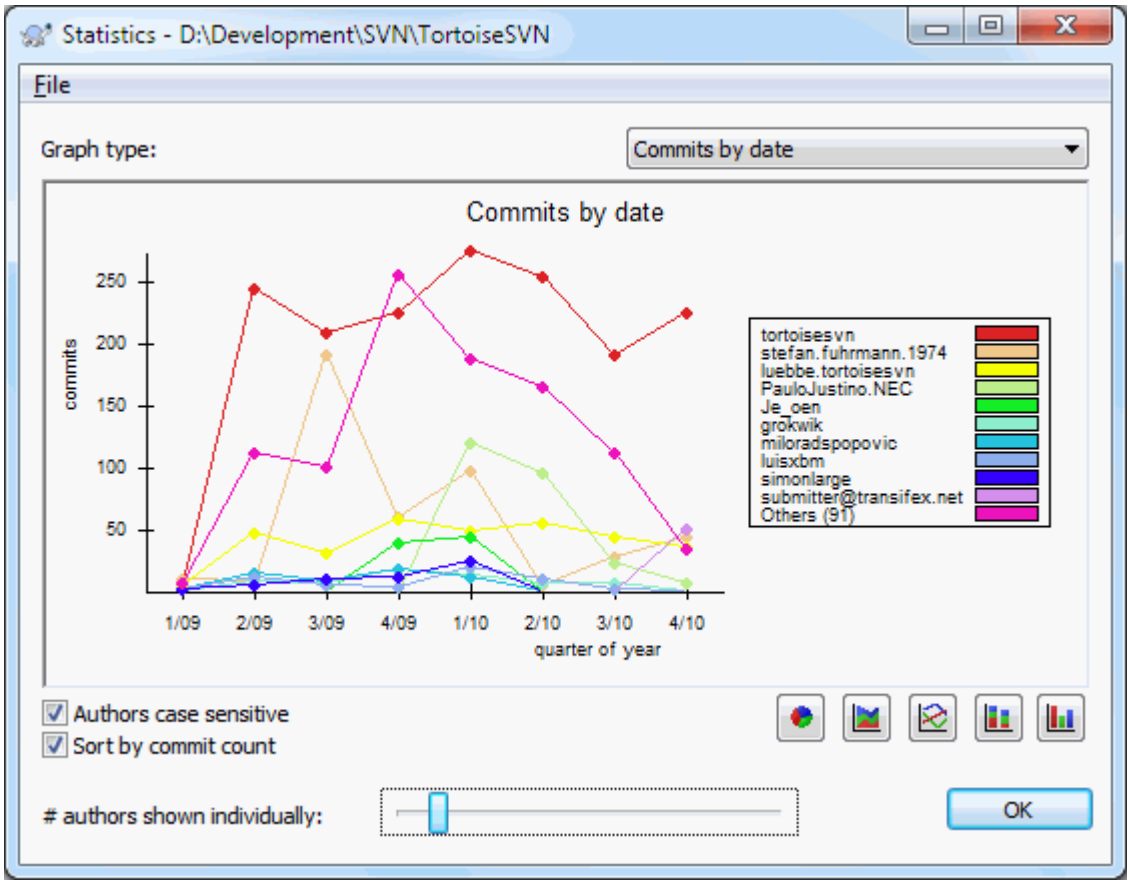
Ta graf prikazuje, kateri avtorji so bili aktivni na projektu v obliki preprostega histograma, zloženega stolpičnega diagrama ali potičnega diagrama.



Slika 4.26. Potični graf objav glede na avtoja

Kadar na projektu sodeluje nekaj glavnih avtorjev in veliko takih, ki prispevajo občasno, veliko majhnih segmentov naredi graf bolj nečitljiv. Drsnik na dnu omogoča nastaviti mejo (v obliki odstotkov). Avtorji, katerih aktivnost je manjša od meje, so združeni v skupino *Ostali*.

4.10.9.3. Objave po datumu



Slika 4.27. Objave po datumu

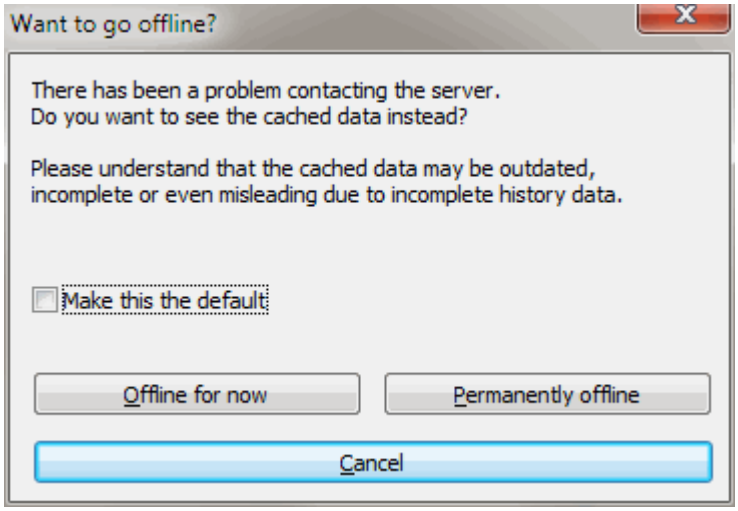
Na tej strani je grafično prikazana projektna aktivnost: število objav *in* avtor. To vam pove, kdaj se na projektu dela in kdo je kdaj delal.

Kadar sodeluje veliko število avtorjev, bo na grafu prikazano veliko število črt. Na voljo sta dva pogleda: *normalen*, kjer je avtorjeva aktivnost relativna na osnovno linijo, in *zložen*, kjer je avtorjeva aktivnost relativna glede na črto pod njim. Ta možnost onemogoča križanje črt, kar olajša branje grafa, vendar je pri tem težje oceniti rezultat posameznega avtorja.

Po privzetih nastavitvah je analiza občutljiva na male in velike črke, torej se uporabnika `PeterEgan` in `PeteRegan` obravnavata kot različna avtorja. Vendar pa v veliko primerih uporabniška imena niso občutljiva na male in velike črke in so zato včasih vnešena nekonsistentno. Zato v takšnih primerih želite imeni `DavidMorgan` in `davidmorgan` obravnavati kot isto osebo. Za nastavitev tega obnašanja uporabite potrditveno polje Za avtorje ni razlikovanja velikih/malih črk.

Statistika pokriva isto časovno območje kot okno dnevnika. Če le-to prikazuje le eno revizijo, potem vam statistika ne pove prav veliko.

4.10.10. Nepovezan način



Slika 4.28. Go Offline Dialog

If the server is not reachable, and you have log caching enabled you can use the log dialog and revision graph in offline mode. This uses data from the cache, which allows you to continue working although the information may not be up-to-date or even complete.

Here you have three options:

Offline for now
: Complete the current operation in offline mode, but retry the repository next time log data is requested.

Permanently offline
: Remain in offline mode until a repository check is specifically requested. See Razdelek 4.10.11, “Osveževanje pogleda”.

Prekliči
: If you don't want to continue the operation with possibly stale data, just cancel.

The Make this the default checkbox prevents this dialog from re-appearing and always picks the option you choose next. You can still change (or remove) the default after doing this from TortoiseSVN → Settings.

4.10.11. Osveževanje pogleda

Če želite preveriti, ali obstajajo na strežniku novejši zapisi v dnevniku, lahko pogled osvežite z uporabe tipke **F5**. Če uporabljate predpomnilnik dnevnika (privzeta nastavitev), s tem preverite skladišče in prenesete le najnovejše zapise. Če ste pred delali v nepovezanem načinu, bo TortoiseSVN poskušal vzpostaviti povezavo.

Če uporabljate predpomnilnik dnevnika in menite, da se je sporočilo dnevniškega zapisa ali avtor spremenil, lahko za ponoven prenos prikazanih sporočil iz strežnika in posodobitev predpomnilnika uporabite **Shift-F5** ali **Ctrl-F5**. Upoštevajte, da s tem osvežite le trenutno prikazana sporočila in ne celotnega predpomnilnika za trenutno skladišče.

4.11. Pregledovanje razlik

Ena izmed najbolj pogostih zahtev v razvoju projekta je videti spremembe. Mogoče želite videti razlike med dvema revizijama iste datoteke ali razlike med dvema različnima datotekama. TortoiseSVN vsebuje orodje TortoiseMerge za pregledovanje razlik besedilnih datotek. Za pregledovanje razlik grafičnih datotek TortoiseSVN ponuja TortoiseIDiff. Seveda pa lahko uporabite tudi programe po lastni izbiri.

4.11.1. Spremembe v datoteki

Krajevne spremembe
: Če želite videti spremembe, ki ste jih naredili *vi*, uporabite kontekstni meni v Raziskovalcu in izberite TortoiseSVN → Razlikuj.

Razlike v primerjavi z drugo vejo/oznako
: Če želite videti, kaj se je spremenilo na glavni veji (če delate na stranski veji) ali na določeni veji (če delate na glavni veji), lahko uporabite kontekstni meni Raziskovalca. Med desnim klikom na datoteko držite pritisnjeno tipko **Shift**. Potem izberite TortoiseSVN → Razlikuj z URL. V okno vnesite naslov URL skladišča, s katerim želite primerjati krajevno datoteko.

  Lahko uporabite tudi brskalnik po skladišču in izberete dve drevesi, ki ju želite primerjati, mogoče dve oznaki ali vejo/oznako in glavno vejo. Kontekstni meni vam omogoča primerjanje z ukazom Primerjaj reviziji. Več o tem razloži Razdelek 4.11.3, “Primerjanje map”.

Razlike v primerjavi s prejšnjo revizijo
: Če želite videti razlike med določeno revizijo in delovno kopijo, uporabite Dnevnik, izberite revizijo, ki vas zanima, in iz kontekstnega menija izberite Primerjaj z delovno kopijo.

  Če želite videti razliko med zadnjo objavljeno revizijo in svojo delovno kopijo, ki ni spremenjena, desno kliknite na datoteko. Nato izberite TortoiseSVN → Razlikuj s prejšno verzijo. Izvedla se bo primerjava med revizijo pred zadnjo objavo (ki je zabeležena v vaši delovni kopiji) in delovno osnovo (BASE). Vidite zadnjo spremembo, narejeno na datoteki, ki je ustvarila vašo delovno kopijo. Spremembe, novejše kot je vaša delovna kopija, niso prikazane.

Razlike med dvema prejšnjima razlikama
: Če želite videti razlike med dvema že objavljenima revizijama, uporabite Dnevnik in izberite reviziji, ki ju želite primerjati (uporabite tipko **Ctrl**). Potem iz kontekstnega menija izberite Primerjaj reviziji.

  Če to storite v Dnevniku za mapo, potem pogovorno okno Primerjaj reviziji prikaže seznam spremenjenih datotek v mapi. Več o tem pove Razdelek 4.11.3, “Primerjanje map”.

Vse spremembe, narejeni pri objavi
: Če želite videti spremembe na vseh datotekah v določeni reviziji v enem pogledu, lahko izberete obliko poenotena razlika (oblika popravkov GNU). Ta prikaže le razlike skupaj z nekaj bližnjimi vrsticami. Ta oblika je težje berljiva kot vizuelna, vendar lahko vidite vse spremembe skupaj. Iz Dnevnika izberite revizijo in iz kontekstnega menija zaženite ukaz Prikaži razlike kot poenotene razlike.

Razlika med datotekami
: Če želite videti razlike med dvema različnima datotekama, lahko to storite neposredno iz Razskovalca, tako da izberete obe datoteki (z uporabo gumba **Ctrl**). Potem iz kontekstnega menija Raziskovalca izberite TortoiseSVN → Razlikuj.

  If the files to compare are not located in the same folder, use the command TortoiseSVN → Diff later to mark the first file for diffing, then browse to the second file and use TortoiseSVN → Diff with "path/of/marked/file". To remove the marked file, use the command TortoiseSVN → Diff later again, but hold down the **Ctrl**-modifier while clicking on it.

Razlika med datoteko/mapo v delovni kopiji in na naslovu URL
: Če želite videti razlike med datoteko v delovni kopiji in datoteko v skladišču Subversion, lahko to storite neposredno iz Raziskovalca. Izberite datoteko, pritisnite tipko **Shift** in desno kliknite, da se pojavi kontekstni meni. Izberite TortoiseSVN → Razlikuj z URL. Enako lahko storite na delovni kopiji. TortoiseMerge prikaže razlike enako kot pri prikazu datoteke popravkov - seznam spremenjenih datotek, ki jih lahko pregledujete eno za drugo.

Razlike z informacijami o krivdi
: Če želite poleg razlik videti tudi avtorja, številko revizije in datum sprememb, lahko rezultat razlikovanja in okrivljenja združite. Več o tem si lahko preberete v Razdelek 4.24.2, “Okrivi spremembe”.

Razlika med mapama
: Orodja, ki jih dobite zraven programa TortoiseSVN, ne omogočajo pregledovanja razlik med strukturami map. Če imate zunanje orodje, ki to omogoča, ga lahko uporabite. Razdelek 4.11.6, “Zunanja orodja za razlikovanje/spajanje” pove nekaj o orodjih, ki jih uporabljamo.

Če ste nastavili zunanje orodje za razlikovanje, lahko uporabite **Shift** ob izbiri ukaza Razlikuj in tako poženete zunanje orodje. O nastavljanju zunanjih orodij lahko več preberete v Razdelek 4.31.5, “Nastaviteve zunanjih programov”.

4.11.2. Nastavitev zaključkov vrstic in presledkov

Med delom na projektu včasih zamenjate zaključke vrstic - namesto `CRLF` uporabite `LF`. Mogoče pa spremenite zamikanje v nekem delu datoteke. S tem žal povzročite, da je veliko vrstic označenih kot spremenjenih, čeprav se vsebina vrstic dejansko ni spremenila. Naslednje možnosti vam omogočajo obvladovanje takšnih sprememb pri primerjanju ali nameščanju popravkov. Nastavitve so na voljo v pogovornih oknih Spoji in Okrivi, prav tako pa tudi v nastavitvah programa TortoiseMerge.

Možnost Prezri zaključke vrstic izloči vse spremembe, ki so posledica sprememb zgolj zaradi različnih zaključkov vrstic.

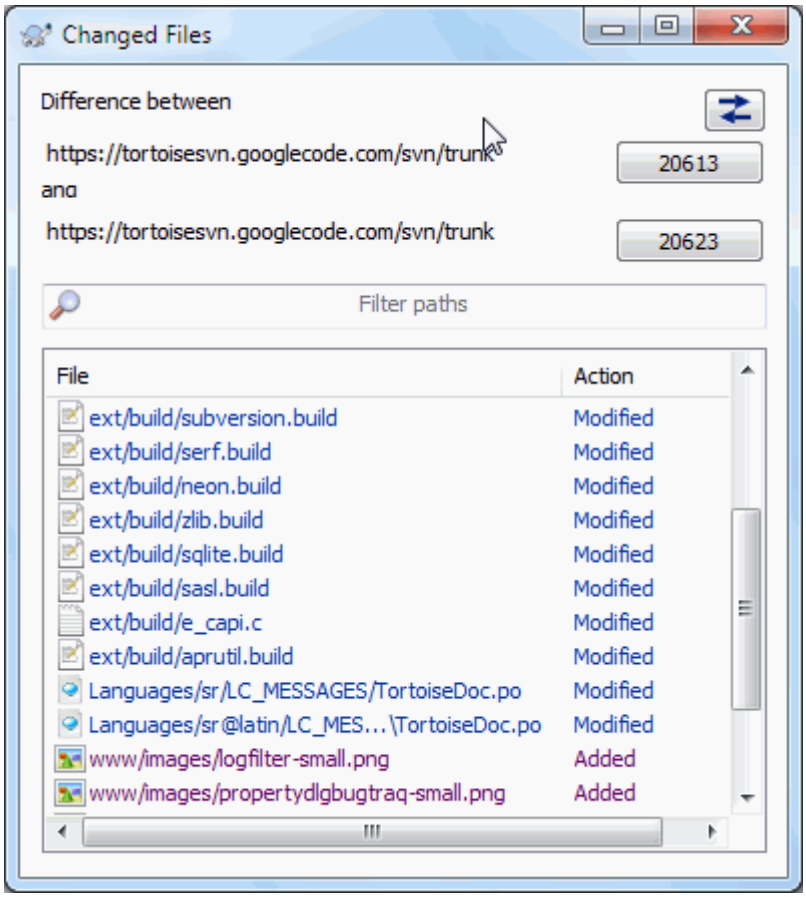
Z možnostjo Primerjaj presledke povzročite, da so vse vrstice, ki vsebuje spremembe presledkov, prikazane kot dodane/odstranjene.

Ignore whitespace changes excludes changes which are due solely to a change in the amount or type of whitespace, e.g. changing the indentation or changing tabs to spaces. Adding whitespace where there was none before, or removing a whitespace completely is still shown as a change.

Možnost Prezri vse presledke izloči vse spremembe, ki so posledica spremembe presledkov.

Seveda so kot spremenjene prikazane vse vrstice s spremenjeno vsebino.

4.11.3. Primerjanje map



Slika 4.29. Okno za primerjanje revizij

Ko v brskalniku po skladišču izberete dve drevesi ali ko izberete dve reviziji mape v dnevniku, lahko uporabite Kontekstni meni → Primerjaj reviziji.

To pogovorno okno prikaže seznam vseh datotek, ki ste jih spremenili, in omogoča posamično primerjavo ali okrivljanje s pomočjo kontekstnega menija.

You can export a *change tree*, which is useful if you need to send someone else your project tree structure, but containing only the files which have changed. This operation works on the selected files only, so you need to select the files of interest - usually that means all of them - and then Context menu → Export selection to.... You will be prompted for a location to save the change tree.

You can also export the *list* of changed files to a text file using Context menu → Save list of selected files to....

If you want to export the list of files *and* the actions (modified, added, deleted) as well, you can do that using Context menu → Copy selection to clipboard.

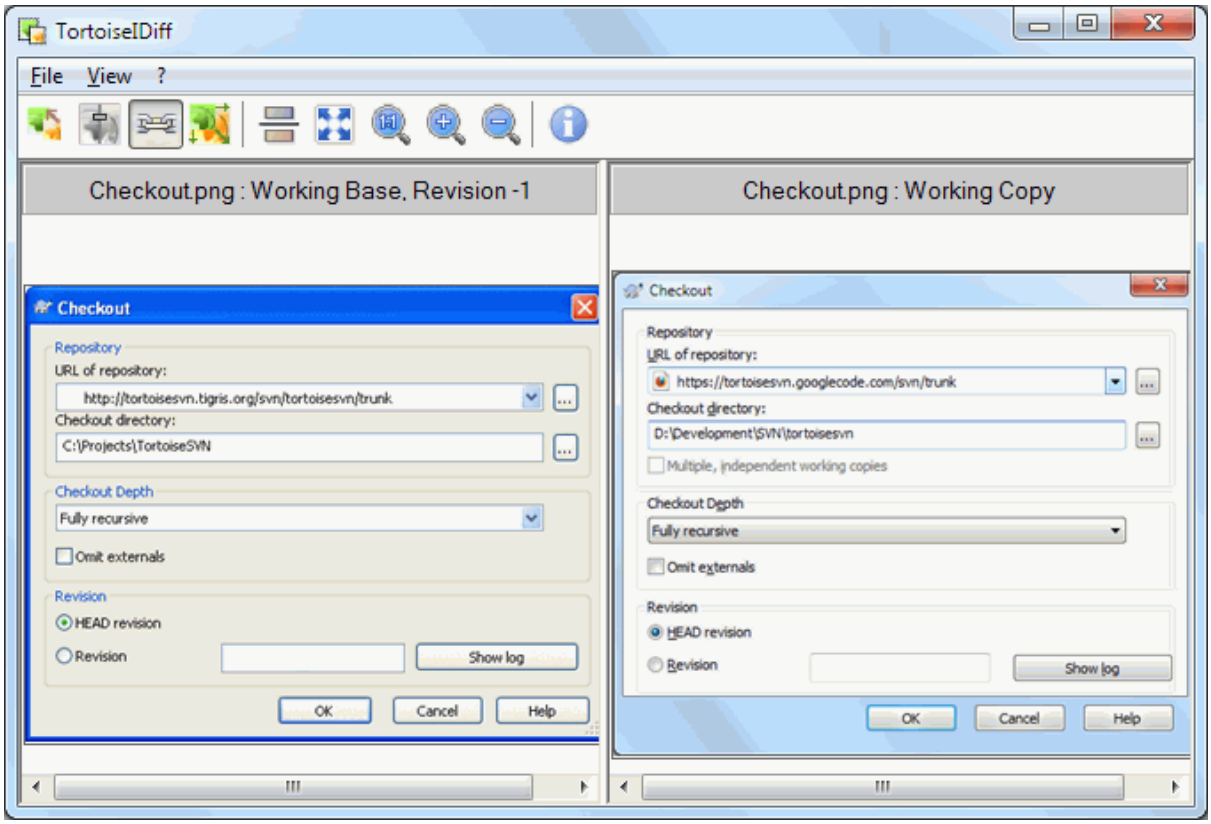
Gumb na vrhu omogoča spremembo smeri primerjave. Lahko prikažete spremembe iz A v B, ali, če vam je lažje, iz B v A.

Gumbi, ki vsebujejo številke revizij, se uporabljajo za spreminjanje območja revizij. Ko območje spremenite, se seznam elementov, ki se razlikujejo med revizijama, samodejno posodobi.

Če je seznam datotek zelo dolg, lahko uporabite polje za iskanje in tako prikažete le datoteke, ki vsebujejo iskane znake. Uporablja se preprosto iskanje, zato za izpis vseh datotek izvorne kode v jeziku C vpišite `.c` in ne `*.c`.

4.11.4. Razlikovanje slik z uporabo programa TortoiseIDiff

Obstaja veliko orodij za razlikovanje besedilnih datotek, vključno z našim lastnim orodjem TortoiseMerge. Pogosto pa želimo videti tudi, kako so se spreminjale slike. Zato smo ustvarili orodje TortoiseIDiff.



Slika 4.30. Pregledovalnik razlik med slikami

Uporaba ukaza TortoiseSVN → Razlikuj na datoteki kateregakoli običajnega slikovnega formata zažene program TortoiseIDiff, ki prikaže razlike med slikama. Po privzetih nastavitvah se sliki prikažeta ena ob drugi, v meniju Pogled pa lahko izberete tudi način prikaza ena slika pod drugo ali celo ena slika čez drugo.

Seveda lahko sliko povečate in pomanjšate ter jo premikate. Premikate lahko tudi s potegom z levim miškinim gumbom. Če izberete možnost Poveži slike, so kontrole za premikanje (drsni trak, kolešček na miški) povezane.

Okno Informacije o sliki prikaže podrobnosti o slikovni datoteki, n. pr. velikost v slikovnih točkah, resolucijo in barvno globino. Če vam je okno v napoto, lahko uporabite Pogled → Informacije o sliki. S tem okno skrijete. Iste informacije se prikažejo v namigu, če z miško mirujete v naslovni vrstici.

Ko se sliki prekrivata, lahko relativno intenzivnost slik (alpha blend) nastavljate z drsnikom na levi strani. Če kliknete kjerkoli na drsniku, boste neposredno nastavili vrednost, lahko pa drsnik potegnete in tako nastavite vrednost interaktivno. Za spreminjanje vrednosti lahko uporabite tudi **Ctrl-Shift**-Zasuk miškinega koleščka.

The button above the slider toggles between 0% and 100% blends, and if you double click the button, the blend toggles automatically every second until you click the button again. This can be useful when looking for multiple small changes.

Včasih želite namesto združene (prekrite) slike raje videti razlike, n. p.r če imate dve reviziji sheme tiskanega vezja in želite izvedeti, katere linije so se spremenile. Če izklopite spajanje (alpha blend) slik, se razlike prikažejo kot *ekskluzivni ALI* barvnih vrednosti točk. Nespremenjena območja bodo bela, spremembe pa bodo obarvane.

4.11.5. Diffing Office Documents

When you want to diff non-text documents you normally have to use the software used to create the document as it understands the file format. For the commonly used Microsoft Office and Open Office suites there is indeed some support for viewing differences and TortoiseSVN includes scripts to invoke these with the right settings when you diff files with the well-known file extensions. You can check which file extensions are supported and add your own by going to TortoiseSVN → Settings and clicking Advanced in the External Programs section.

Problems with Office 2010

If you installed the *Click-to-Run* version of Office 2010 and you try to diff documents you may get an error message from Windows Script Host something like this: “ActiveX component can't create object: word.Application”. It seems you have to use the MSI-based version of Office to get the diff functionality.

4.11.6. Zunanja orodja za razlikovanje/spajanje

Če orodja, ki jih ponujamo v paketu, ne zadostujejo vašim potrebam, lahko poskusite številne odprtokodne ali komercialne programe. Vsakdo ima svoje priljubljeno orodje in ta seznam vsekakor ni popoln, vseeno pa poda nekaj imen aplikacij, ki jih lahko uporabite:

WinMerge
: *WinMerge* [https://winmerge.sourceforge.net/] is a great open-source diff tool which can also handle directories.

Perforce Merge
: Perforce is a commercial RCS, but you can download the diff/merge tool for free. Get more information from *Perforce* [https://www.perforce.com/perforce/products/merge.html].

KDiff3
: KDiff3 je brezplačno orodje za razlikovanje, ki zna delati z mapami. Dobite ga *tukaj* [http://kdiff3.sf.net/].

SourceGear DiffMerge
: SourceGear Vault is a commercial RCS, but you can download the diff/merge tool for free. Get more information from *SourceGear* [https://www.sourcegear.com/diffmerge/].

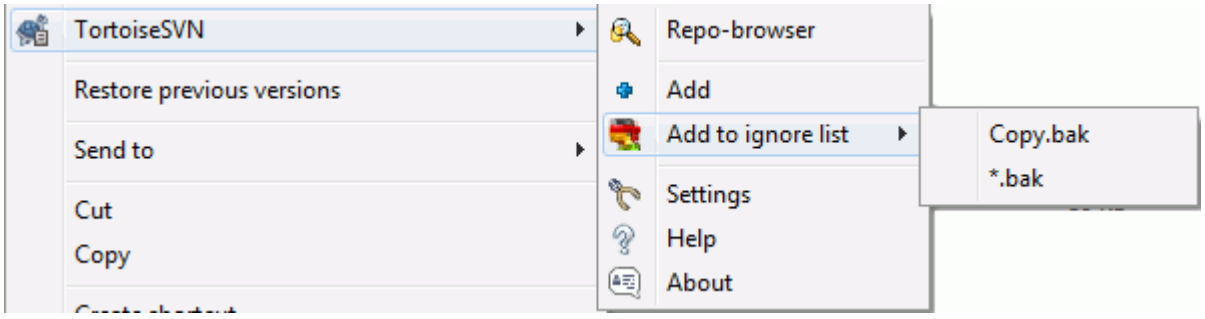
ExamDiff
: ExamDiff Standard je brezplačen progam. Zna delati z datotekami, ne pa tudi z mapami. ExamDiff Pro je preizkusni program (shareware) in ima v primerjavi z brezplačno različico veliko dodatkov, vključno z možnostjo razlikovanja in urejanja map. Od verzije 3.2 naprej znata obe različici delati z datotekami tipa unicode. Na voljo sta na strani podjetja *PrestoSoft* [http://www.prestosoft.com/].

Beyond Compare
: Similar to ExamDiff Pro, this is an excellent shareware diff tool which can handle directory diffs and unicode. Download it from *Scooter Software* [https://www.scootersoftware.com/].

Araxis Merge
: Araxis Merge is a useful commercial tool for diff and merging both files and folders. It does three-way comparison in merges and has synchronization links to use if you've changed the order of functions. Download it from *Araxis* [https://www.araxis.com/merge/index.html].

Preberite Razdelek 4.31.5, “Nastaviteve zunanjih programov”, če želite izvedeti več o uporabi teh orodij.

4.12. Dodajanje novih datotek in map



Slika 4.31. Kontekstni menu v Raziskovalcu za datoteke brez različic

Če ste med razvojem ustvarili nove datoteke in/ali mape, jih morate dodati v nadzor različic. Izberite datoteke in map in uporabite ukaz TortoiseSVN → Dodaj.

Potem ko ste datoteke/mape dodali v nadzor različic, so označene z ikono `dodano`, kar pomeni, da morate delovno kopijo objaviti, preden so dodane datoteke/mape na razplago ostalim uporabnikom. Zgolj dodajanje datoteke/mape *ne* vpliva na skladišče!

Veliko dodanih elementov

Ukaz Dodaj lahko uporabite tudi na mapi, ki je že pod nadzorom različic. V tem primeru se bodo v pogovornem oknu za dodajanje pojavile vse datoteke brez različic. To vam pomaga, če imate veliko novih datotek, ki jih želite dodati naenkrat.

Če želite dodati datoteke, ki se ne nahajajo v delovni kopiji, uporabite metodo povleci in spusti:

1. izberite datoteke, ki jih želite dodati
2. right drag them to the new location inside the working copy
3. sprostite gumb miške
4. izberite Kontekstni meni → SVN - Dodaj datoteke v to delovno kopijo. S tem datoteke prekopirate v delovno kopijo in jih dodate v nadzor različic.

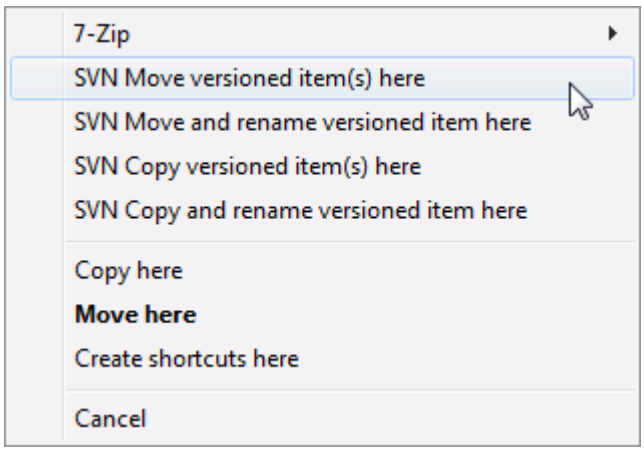
Datoteke, ki se nahajajo v delovni kopiji, lahko dodate tudi tako, da jih povlečete iz Raziskovalca v okno za objave.

Če datoteko ali mapo dodate po pomoti, lahko dodajanje pred objavo prekličete z ukazom TortoiseSVN → Prekliči dodajanje....

4.13. Kopiranje/premikanje/preimenovanje obstoječih datotek in map

It often happens that you already have the files you need in another project in your repository, and you simply want to copy them across. You could simply copy the files and add them, but that would not give you any history. And if you subsequently fix a bug in the original files, you can only merge the fix automatically if the new copy is related to the original in Subversion.

The easiest way to copy files and folders from within a working copy is to use the right drag menu. When you right drag a file or folder from one working copy to another, or even within the same folder, a context menu appears when you release the mouse.



Slika 4.32. Meni ob premikanju mape, ki je pod nadzorom različic

Now you can copy existing versioned content to a new location, possibly renaming it at the same time.

You can also copy or move versioned files within a working copy, or between two working copies, using the familiar cut-and-paste method. Use the standard Windows Copy or Cut to copy one or more versioned items to the clipboard. If the clipboard contains such versioned items, you can then use TortoiseSVN → Paste (note: not the standard Windows Paste) to copy or move those items to the new working copy location.

Datoteke in mape lahko prekopirate na drugo lokacijo v skladišču z uporabo ukaza TortoiseSVN → Veja/oznaka. Preberite Razdelek 4.20.1, “Ustvarjanje veje ali oznake”.

Starejšo različico datoteke ali mape lahko poiščete v dnevniku in jo skopirate na novo lokacijo v skladišču z uporabo ukaza Kontekstni meni → Ustvari vejo/oznako iz revizije. Za več informacij preberite Razdelek 4.10.3, “Pridobivanje dodatnih informacij”.

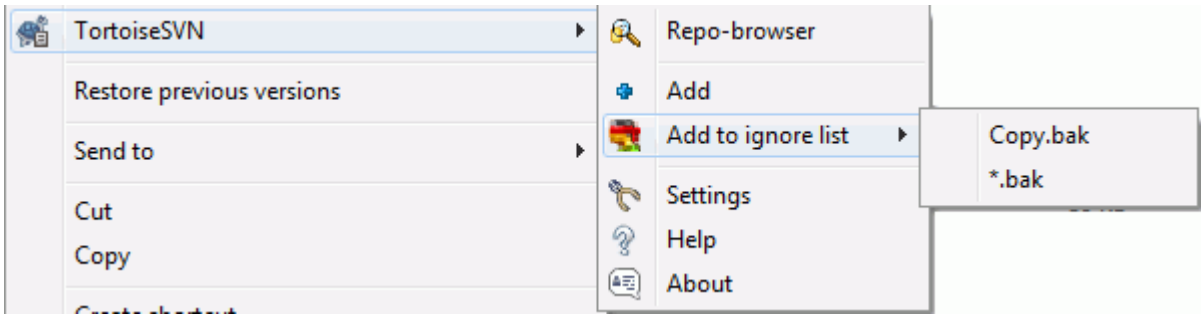
Brskalnik po skladišču pa lahko uporabite tudi za iskanje željene vsebine, ki jo lahko prekopirate v delovno kopijo neposredno iz skladišča ali pa jo kopirate med dvema lokacijama v skladišču. Za več informacij preberite Razdelek 4.25, “Brskalnik po skladišču”.

Kopiranje med skladišči ni mogoče

Whilst you can copy or move files and folders *within* a repository, you *cannot* copy or move from one repository to another while preserving history using TortoiseSVN. Not even if the repositories live on the same server. All you can do is copy the content in its current state and add it as new content to the second repository.

Če niste prepričani, ali dva naslova URL na istem strežniku kažeta na isto skladišče ali na dve različni skladišči, uporabite brskalnik po skladišču. Odprite prvi naslov URL in poiščite njegov koren. Če v enem brskalniku po skladišču vidite obe lokaciji, se obe lokaciji nahajata v istem skladišču.

4.14. Dodajanje datotek in map na seznam prezrtih elementov



Slika 4.33. Kontekstni menu v Raziskovalcu za datoteke brez različic

V večini projektov obstajajo datoteke in mape, ki jih ne dajemo pod nadzor različic. To so datoteke, ki jih naredi prevajalnik (`*.obj, *.lst`), mape, kamor se shranjujejo izvršne datoteke. Ko objavljate spremembe, TortoiseSVN prikaže datoteke brez različic, kar napolni seznam v oknu za objave. Seveda lahko prikaz datotek brez različic izklopite, vendar lahko to povzroči, da pozabite dodati nove izvorne datoteke v skladišče.

Tem problemom se najlažje izognete, če dodate generirane datoteke na projektni seznam prezrtih elementov. Na ta način se te datoteke ne bodo pojavile v oknu za objave, medtem ko boste tam še vedno videli izvorne datoteke brez različic.

If you right click on a single unversioned file, and select the command TortoiseSVN → Add to Ignore List from the context menu, a submenu appears allowing you to select just that file, or all files with the same extension.

Both submenus also have a `(recursively)` equivalent. If you select multiple files, there is no submenu and you can only add those specific files/folders.

If you choose the `(recursively)` version of the ignore context menu, the item will be ignored not just for the selected folder but all subfolders as well. However this requires SVN clients version 1.8 or higher.

Če želite iz seznama prezrtih odstraniti enega ali več elementov, desno kliknite na elemente in izberite TortoiseSVN → Odstrani s seznama prezrtih Dostopate pa lahko tudi neposredno do lastnosti `svn:ignore` določene mape. To vam omogoča, da določite bolj splošne vzorce za preziranje, kar je opisano spodaj. Za neposredno nastavljanje lastnosti preberite Razdelek 4.18, “Nastavitve projekta”. Upoštevajte, da je vsak vzorec za prezrte datoteke potrebno vpisati v svojo vrstico. Ločevanje s presledki ne deluje.

Splošni seznam prezrtih elementov

Drug način, kako prezreti datoteke, je dodajanje na *splošni seznam prezrtih elementov*. Razlika je v tem, da je splošen seznam lastnost odjemalca. Velja za *vse* projekte v sistemu Subversion, vendar le na računalniku, kjer je nameščen odjemalec. Kjer je to le mogoče, je v splošnem bolje uporabiti lastnost `svn:ignore`, ker jo lahko nastavimo na določene dele projekta, poleg tega pa velja za vse uporabnike, ki prevzamejo projekt. Za več informacij preberite Razdelek 4.31.1, “Splošne nastavitve”.

Dodajanje elementov pod nadzorom različic na seznam prezrtih

Datotek in map pod nadzorom različic ni mogoče prezreti. Če ste dali datoteko pod nadzor različic po pomoti, preberite Razdelek B.8, “Kako dodam na seznam prezrtih datoteke, ki so že pod nadzorom”, kjer je opisano, kako jo iz sistema nadzora “odstranite”.

4.14.1. Iskanje vzorcev v seznamu prezrtih elementov

Iskanje vzorcev v sistemu Subversion uporablja tehniko, ki izvira iz sistema Unix in se uporablja za določanje datotek z uporabo meta-znakov. Naslednji znaki imajo poseben pomen:

*
: Poišče katerikoli niz znakov ali prazen niz (brez znakov).

?
: Poišče katerikoli znak.

[...]
: Poišče katerikoli znak, ki je podan v oglatem oklepaju. Par znakov znotraj oklepajev, ločen z “-”, pomeni katerikoli znak med tema dvema znakoma. Primer: `[AGm-p]` išče kateregakoli izmed naslednjih znakov: `A`, `G`, `m`, `n`, `o` ali `p`.

Pattern matching is case sensitive, which can cause problems on Windows. You can force case insensitivity the hard way by pairing characters, e.g. to ignore `*.tmp` regardless of case, you could use a pattern like `*.[Tt][Mm][Pp]`.

Če vas zanima uradna definicija iskanja vzorcev v datotekah, jo najdete v Specifikaciji IEEE za ukaze lupine *Pattern Matching Notation* [http://www.opengroup.org/onlinepubs/009695399/utilities/xcu_chap02.html#tag_02_13].

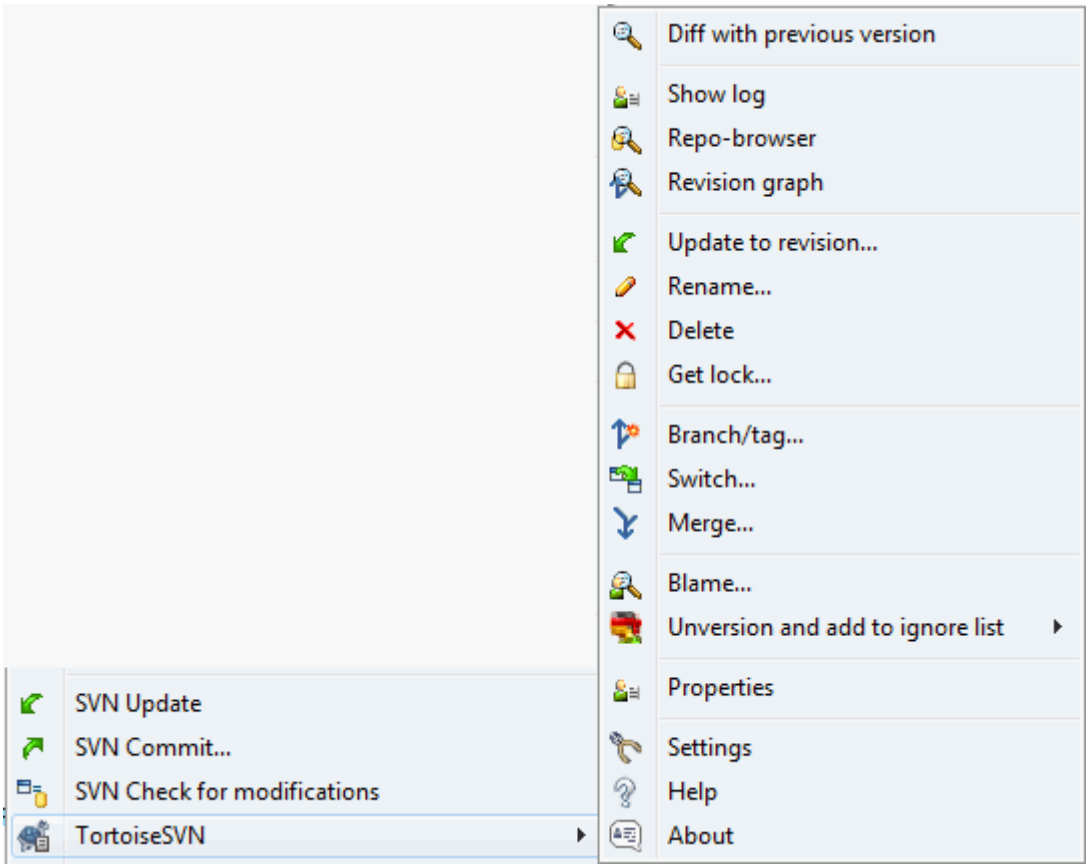
No Paths in Global Ignore List

You should not include path information in your pattern. The pattern matching is intended to be used against plain file names and folder names. If you want to ignore all `CVS` folders, just add `CVS` to

> the ignore list. There is no need to specify `CVS */CVS` as you did in earlier versions. If you want to ignore all `tmp` folders when they exist within a `prog` folder but not within a `doc` folder you should use the `svn:ignore` property instead. There is no reliable way to achieve this using global ignore patterns.

4.15. Brisanje, preimenovanje in premikanje

Subversion allows renaming and moving of files and folders. So there are menu entries for delete and rename in the TortoiseSVN submenu.



Slika 4.34. Konteksni meni v Raziskovalcu za datotek pod nadzorom različic

4.15.1. Brisanje datotek in map

Use TortoiseSVN → Delete to remove files or folders from Subversion.

When you TortoiseSVN → Delete a file or folder, it is removed from your working copy immediately as well as being marked for deletion in the repository on next commit. The item's parent folder shows a “modified” icon overlay. Up until you commit the change, you can get the file back using TortoiseSVN → Revert on the parent folder.

Če želite izbrisati element iz skladišča, vendar ga želite obdržati krajevno kot datoteko/mapo brez različic, uporabite ukaz Razširjeni kontekstni meni → Izbriši (ohrani krajevno). Za prikaz razširjenega kontekstnega menija morate držati tipko **Shift** ob desnem kliku na element v Raziskovalcu.

If an item is deleted via the explorer instead of using the TortoiseSVN context menu, the commit dialog shows those items as missing and lets you remove them from version control too before the commit. However, if you

update your working copy, Subversion will spot the missing item and replace it with the latest version from the repository. If you need to delete a version-controlled file, always use TortoiseSVN → Delete so that Subversion doesn't have to guess what you really want to do.

Kako dobiti nazaj izbrisane datoteke ali mape

If you have deleted a file or a folder and already committed that delete operation to the repository, then a normal TortoiseSVN → Revert can't bring it back anymore. But the file or folder is not lost at all. If you know the revision the file or folder got deleted (if you don't, use the log dialog to find out) open the repository browser and switch to that revision. Then select the file or folder you deleted, right click and select Context Menu → Copy to... as the target for that copy operation select the path to your working copy.

4.15.2. Premikanje datotek in map

Če želite datoteko preimenovati, uporabite Kontekstni meni → Preimenuj... Vpišite novo ime za element in preimenovanje je opravljeno.

If you want to move files around inside your working copy, perhaps to a different sub-folder, use the right mouse drag-and-drop handler:

1. izberite datoteke ali mape, ki jih želite premakniti
2. right drag them to the new location inside the working copy
3. sprostite gumb miške
4. v meniju izberite Kontekstni meni → SVN - Premakni datoteke pod nadzorom sem

Objavi nadrejeno mapo

Ker je preimenovanje izvedeno kot zaporedni operaciji brisanja in dodajanja, morate narediti objavo na nadrejeni mapi datoteke, da se bo tudi izbrisana datoteka prikazala v oknu za objavo. Če izbrisane datoteke ne objavite, bo le-ta ostala v skladišču in ne bo odstanjena iz delovnih kopij vaših sodelavcev, ko bodo ti posodabljali mapo. V delovni kopiji bodo *obe* datoteki - tako novo kot staro.

Preimenovanje mape *morate* objaviti preden začnete spreminjati njeno vsebino, v nasprotnem primeru lahko lahko pride do okvare delovne kopije.

Another way of moving or copying files is to use the Windows copy/cut commands. Select the files you want to copy, right click and choose Context Menu → Copy from the explorer context menu. Then browse to the target folder, right click and choose TortoiseSVN → Paste. For moving files, choose Context Menu → Cut instead of Context Menu → Copy.

Za premikanje elementov lahko uporabite tudi brskalnik po skladišču. Za več informacij preberite Razdelek 4.25, “Brskalnik po skladišču”.

Zunanjih elementov ne premikajte z ukazi Subversion

Ukazov Premakni in Preimenuj *ne* uporabljajte na mapah, ki imajo nastavljeno lastnost `svn:externals`. To bi povzročilo izbris zunanjega elementa iz nadrejene mape, kar bi najbrž razjezilo precej uporabnikov. Če želite premakniti zunanjo mapo, uporabite premikanje v Raziskovalcu, nato pa uskladite lastnosti `svn:externals` nadrejenih map vira in cilja.

4.15.3. Obravnava težav pri imenih datotek zaradi velikih in malih črk

Če se v skladišču že nahajata dve datoteki, ki se razlikujeta samo v velikih in malih črkah (n. pr. `TEST.TXT` in `test.txt`), mape, kjer se ti dve datoteki nahajata, na sistemu Windows ne morete posodobiti ali prevzeti. Subversion namreč omogoča ločevanje imen, ki se razlikujejo samo v velikih in malih črkah, sistem Windows pa ne.

Ta situacija je možna, če dva uporabnika iz dveh različnih delovnih kopij objavita datoteko, ki se razlikuje le v velikih in malih črkah. Možna pa je tudi takrat, ko uporabniki uporabljajo operacijski sistem, ki v datotečnem sistemu ločuje velike in male črke, n. pr. Linux.

V takšnem primeru se morate odločiti, katero izmed dveh datotek boste obdržali, drugo pa morate v skladišču izbrisati ali premenovati.

Kako preprečiti dve datoteki z enakima imenoma

There is a server hook script available at: https://svn.apache.org/repos/asf/subversion/trunk/contrib/hook-scripts/ that will prevent checkins which result in case conflicts.

4.15.4. Popravljanje preimenovanj datotek

Včasih vaše prijazno razvojno orodje preimenuje datoteke kot del refaktoriranja in seveda tega ne pove sistemu Subversion. Če poskusite takšne spremembe objaviti, vidite staro datoteko kot manjkajočo in novo kot datoteko brez različic. Lahko bi enostavno dodali datoteko brez različic, vendar bi v tem primeru izgubili zgodovino, saj Subversion ne ve, da sta datoteki povezani.

Boljši način je, da sistemu Subversion poveste, da gre v tem primeru dejansko za preimenovanje datoteke in sicer lahko to storite v oknih Objavi in Preveri posodobitve. Enostavno izberite tako staro datoteko (manjkajočo) kot novo (brez različic) in uporabite Kontekstni meni → Popravi premik. S tem datoteki povežete v preimenovanje.

4.15.5. Brisanje datotek brez različic

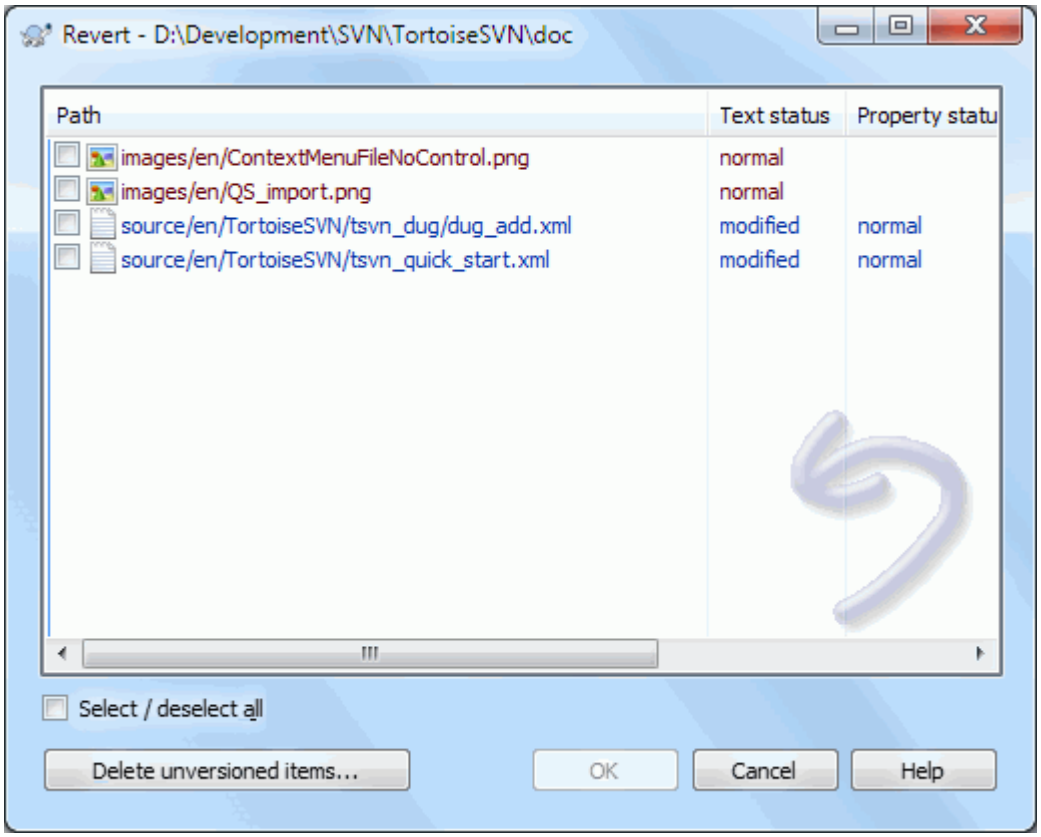
Običajno nastavite sezname prezrtih elementov v sistemu v Subversion tako, da se prezrejo vse generirane datoteke. Kaj pa, če želite počistiti vse prezrte datoteke, da bi naredili čisto gradnjo? Običajno naredite to v skripti makefile, če pa to skripto razhroščujete ali delate spremembe v sistemu gradnje, je koristno, če lahko delovno kopijo počistite.

TortoiseSVN vam ponuja to možnost z uporabo ukaza Razširjeni kontekstni meni → Izbriši datoteke brez različic.... Da vidite razširjeni kontekstni meni, med desnim klikom na mapo v Raziskovalcu držite pritisnjeno tipko **Shift**. Prikazalo se bo pogovorno okno s seznamom vseh datotek brez različic v delovni kopiji. Izberete lahko tiste, ki jih želite izbrisati.

Pri brisanju elementov se le-te premakne v Koš, tako da jih lahko v primeru napake še vedno pridobite nazaj.

4.16. Razveljavljanje sprememb

Če želite v datoteki povrniti vse spremembe, ki ste jih naredili od zadnje posodobitve, desno kliknite, da se pojavi kontekstni meni in izberite ukaz TortoiseSVN → Povrni Pojavi se pogovorno okno s seznamom vseh spremenjenih datotek, ki jih lahko povrnete. Izberite tiste, ki jih želite povrniti in kliknite na gumb V redu.



Slika 4.35. Okno za povrnitev

If you also want to clear all the changelists that are set, check the box at the bottom of the dialog.

If you want to undo a deletion or a rename, you need to use Revert on the parent folder as the deleted item does not exist for you to right click on.

Če želite preklicati dodajanje elementa, lahko iz kontekstnega menija izberete ukaz TortoiseSVN → Prekliči dodajanje.... Dejansko se izvede ukaz Povrni, ki pa smo ga preimenovali, da je njegovo dejanje bolj očitno.

Stolpce v tem oknu lahko prilagodite enako kot stolpce v oknu Preveri spremembe. Za več informacij preberite Razdelek 4.7.3, “Krajevno in oddaljeno stanje”.

Since revert is sometimes used to clean up a working copy, there is an extra button which allows you to delete unversioned items as well. When you click this button another dialog comes up listing all the unversioned items, which you can then select for deletion.

Povrnitev sprememb, ki so že bile objavljene

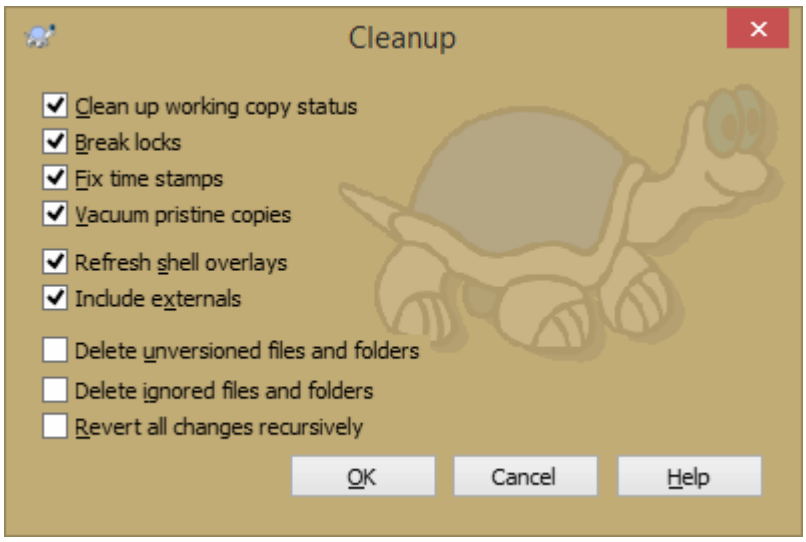
Z ukazom Povrni lahko povrnete le spremembe, ki jih še niste objavili, *ne* pa tudi sprememb, ki ste jih že objavili. Če želite povrniti vse spremembe, ki ste jih objavili v določeni reviziji, preberite Razdelek 4.10, “Pogovorno okno Dnevnik”.

Povrnitev je počasna

Povrnitev krajevnih sprememb lahko traja dalj časa, kot mogoče pričakujete. To se zgodi, ker TortoiseSVN spremenjeno različico datoteke pošlje v Koš, tako da jo lahko pridobite, če ste ukaz Povrni izvedli po pomoti. Če je Koš poln, potrebuje operacijski sistem več časa, da najde prostor, kamor datoteko shrani. Rešitev je enostavna: izpraznite Koš ali pa v nastavitvah izklopite polje Pri uporabi ukaza Povrni uporabi koš.

4.17. Čiščenje

If a Subversion command cannot complete successfully, perhaps due to server problems, your working copy can be left in an inconsistent state. In that case you need to use TortoiseSVN → Cleanup on the folder. It is a good idea to do this at the top level of the working copy.



Slika 4.36. The Cleanup dialog

In the cleanup dialog, there are also other useful options to get the working copy into a `clean` state.

Clean up working copy status
: As stated above, this option tries to get an inconsistent working copy into a workable and usable state. This doesn't affect any data you have but only the internal states of the working copy database. This is the actual `Cleanup` command you know from older TortoiseSVN clients or other SVN clients.

Break write locks
: If checked, all write locks are removed from the working copy database. For most situations, this is required for the cleanup to work!

  Only uncheck this option if the working copy is used by other users/clients at the time. But if the cleanup then fails, you have to check this option for the cleanup to succeed.

Fix time stamps
: Changes all file times to the time of the last commit.

Vacuum pristine copies
: Removes unused pristine copies and compresses all remaining pristine copies of working copy files.

Refresh shell overlays
: Sometimes the shell overlays, especially on the tree view on the left side of the explorer don't show the current status, or the status cache failed to recognize changes. In this situation, you can use this command to force a refresh.

Include externals
: If this is checked, then all actions are done for all files and folders included with the `svn:externals` property as well.

Delete unversioned files and folders, Delete ignored files and folders
: This is a fast and easy way to remove all generated files in your working copy. All files and folders that are not versioned are moved to the trash bin.

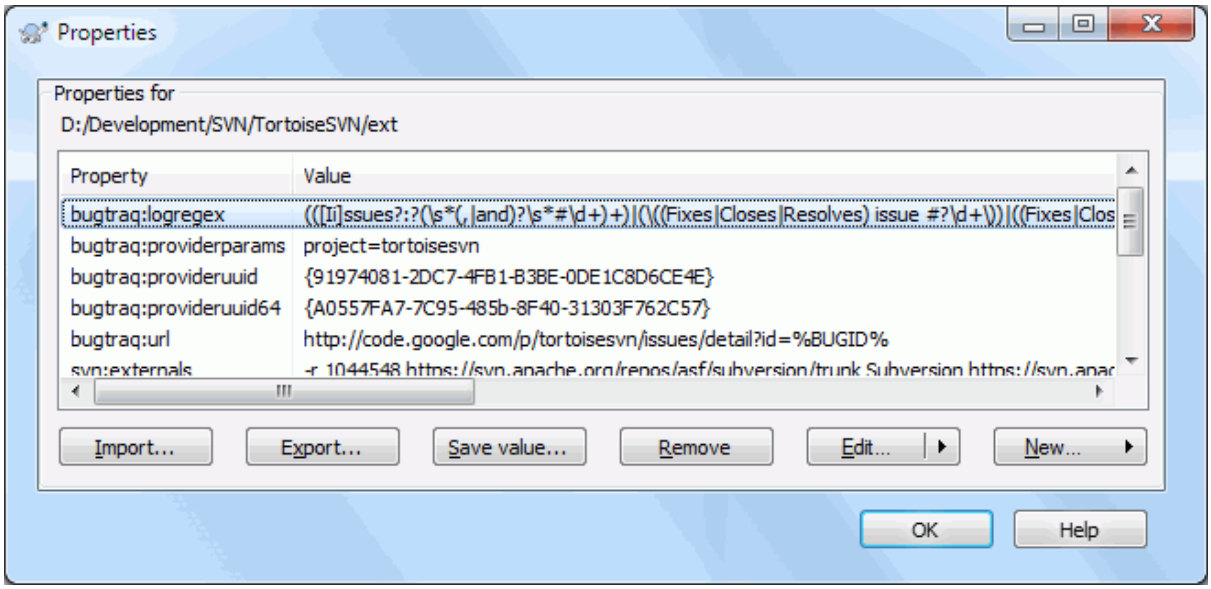
Note: you can also do the same from the TortoiseSVN → Revert dialog. There you also get a list of all the unversioned files and folders to select for removal.

Revert all changes recursively
This command reverts all your local modifications which are not committed yet.

Note: it's better to use the TortoiseSVN → Revert command instead, because there you can first see and select the files which you want to revert.

4.18. Nastavitve projekta

4.18.1. Lastnosti Subversion



Slika 4.37. Lastnosti v sistemu Subversion

Lastnosti Subversion lahko preberete in nastavite v oknu Lastnosti, ki ga prikaže sistem Windows. Poleg tega vidite te lastnosti tudi, če uporabite ukaz TortoiseSVN → Lastnosti. V pogovornih oknih TortoiseSVN, kjer se izpišejo seznami datotek, lahko lastnosti prikažete z ukazom Kontekstni meni → Lastnosti.

You can add your own properties, or some properties with a special meaning in Subversion. These begin with `svn:`. `svn:externals` is such a property; see how to handle externals in Razdelek 4.19, “External Items”.

4.18.1.1. svn:keywords

Subversion omogoča (podobno kot sistem CVS) razširitev ključnih besed, ki se lahko uporabljajo za vstavljanje imena datoteke in številke revizije v vsebino datoteke. Trenutno so podprte naslednje ključne besede:

$Date$
Datum zadnje znane objave glede na informacije iz zadnje posodobitve delovne kopije. *Ne* preverja, če so v skladišču novejše spremembe.

$Revision$
Revizija zadnje znane objave.

$Author$
Avtor zadnje znane objave.

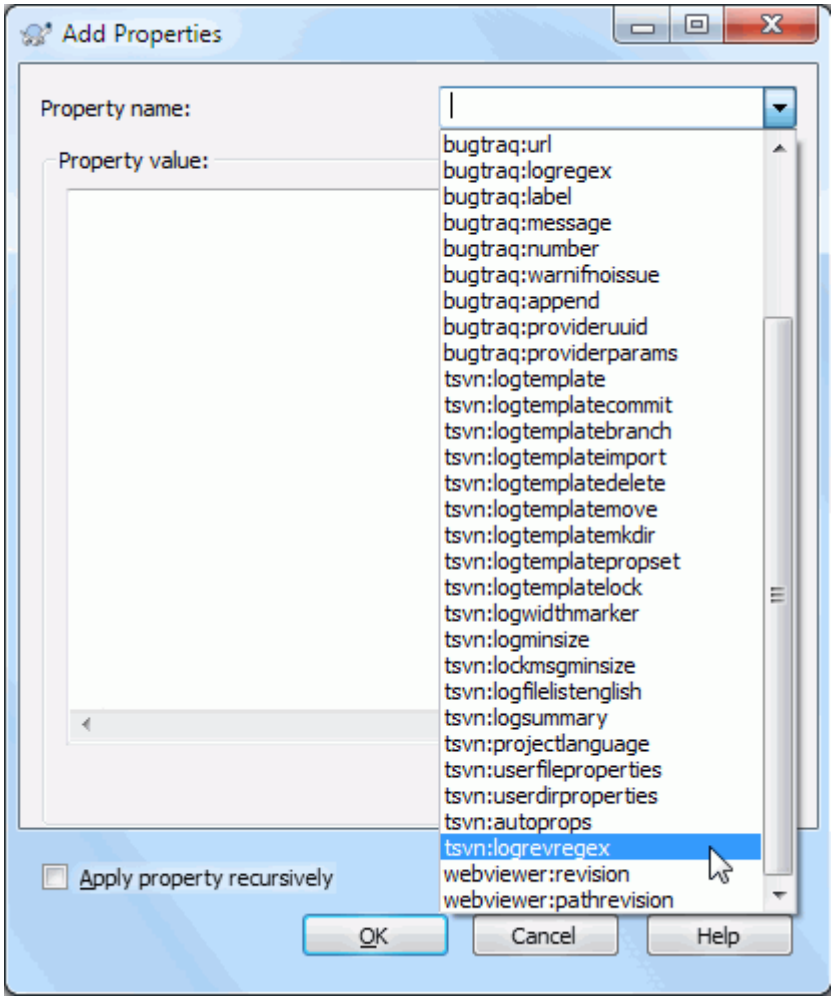
$HeadURL$
: Celoten naslov URL te datoteke v skladišču.

Id
: Stisnjena kombinacija prejšnjih štirih ključnih besed.

To find out how to use these keywords, look at the *svn:keywords section* [http://svnbook.red-bean.com/en/1.8/svn.advanced.props.special.keywords.html] in the Subversion book, which gives a full description of these keywords and how to enable and use them.

For more information about properties in Subversion see the *Special Properties* [http://svnbook.red-bean.com/en/1.8/svn.advanced.props.html].

4.18.1.2. Dodajanje in urejanje lastnosti



Slika 4.38. Dodajanje lastnosti

To add a new property, first click on New.... Select the required property name from the menu, and then fill in the required information in the specific property dialog. These specific property dialogs are described in more detail in Razdelek 4.18.3, “Property Editors”.

To add a property that doesn't have its own dialog, choose Advanced from the New... menu. Then either select an existing property in the combo box or enter a custom property name.

If you want to apply a property to many items at once, select the files/folders in explorer, then select Context menu → properties.

Če želite nastaviti lastnost *vsaki* datoteki in mapi v hierarhiji pod trenutno mapo, uporabite potrditveno polje Rekurzivno.

Če želite urediti obstoječo lastnost, jo izberite iz seznama obstoječih lastnosti in kliknite na gumb Uredi....

Če želite odstraniti nastavljeno lastnost, jo izberite iz seznama obstoječih lastnosti in kliknite na gumb Odstrani.

The `svn:externals` property can be used to pull in other projects from the same repository or a completely different repository. For more information, read Razdelek 4.19, “External Items”.

Edit properties at HEAD revision

Because properties are versioned, you cannot edit the properties of previous revisions. If you look at properties from the log dialog, or from a non-HEAD revision in the repository browser, you will see a list of properties and values, but no edit controls.

4.18.1.3. Izvažanje in uvažanje lastnosti

Pogosto isti nabor lastnosti nastavite na več datotekah/mapah, naprimer `bugtraq:logregex`. Za enostaven prenos lastnosti iz enega projekta na drugega lahko uporabite zmožnost Izvoz/Uvoz.

Na datoteki ali mapi, ki vsebuje nastavljene lastnosti, uporabite TortoiseSVN → Lastnosti, izberite lastnosti, ki jih želite izvoziti in kliknite na gumb Izvozi.... Vpisati boste morali ime datoteke, v katero naj se lastnosti shranijo.

Na mapah, na katerih želite nastaviti lastnosti, uporabite TortoiseSVN → Lastnosti in kliknite na gumb Uvozi.... Izberite datoteko, iz katere želite uvoziti lastnosti, zato izberite datoteko, ki ste jo ustvarili pri izvažanju. Lastnosti bodo v mapo dodane nerekurzivno.

Če želite drevesu dodati lastnosti rekurzivno, naredite zgoraj opisane korake, potem pa v oknu lastnosti izberite vsako lastnost, kliknite na gumb Uredi..., potrdite polje Uporabi lastnost rekurzivno in kliknite na gumb V redu.

Izvozna datoteka je binarna in lastna sistemu TortoiseSVN. Njena edina uporaba je prenos lastnosti z uporabo ukazov Izvozi in Uvozi, tako da teh datotek ne urejamo.

4.18.1.4. Binarne lastnosti

TortoiseSVN lahko z uporabo datotek dodaja lastnostim dvojiške vrednosti. Če želite dvojiško lastnost prebrati, jo shranite v datoteko z gumbom Shrani.... Če želite dvojiško lastnost nastaviti, uporabite urejevalnik datotek ali drugo primerno orodje, ki vam omogoča ustvariti datoteko z željeno vsebino, potem pa datoteko naložite s klikom na gumb Naloži....

Čeprav se dvojiške lastnosti ne uporabljajo prav pogosto, pa so včasih precej uporabne. Primer: če shranjujete obsežne grafične datoteke ali če je aplikacija, ki slike nalaga, obsežna, lahko pomanjšane slike shranite kot lastnosti datotek in jih tako hitreje pregledujete.

4.18.1.5. Samodejna nastavitev lastnosti

Subversion in TortoiseSVN lahko prilagodite tako, da se lastnosti samodejno nastavijo datotekam in mapam, ko se te dodajo v skladišče. To lahko storite na dva načina.

You can edit the Subversion configuration file to enable this feature on your client. The General page of TortoiseSVN's settings dialog has an edit button to take you there directly. The config file is a simple text file which controls some of Subversion's workings. You need to change two things: firstly in the section headed `miscellany` uncomment the line `enable-auto-props = yes`. Secondly you need to edit the section below to define which properties you want added to which file types. This method is a standard Subversion feature and works with any Subversion client. However it has to be defined on each client individually - there is no way to propagate these settings from the repository.

Druga možnost je nastavitev lastnosti `tsvn:autoprops` na mapah, kot je opisano v naslednjem odseku. Ta metoda deluje le pri odjemalcih TortoiseSVN, vendar se pri posodobitvi prenese na vse delovne kopije.

As of Subversion 1.8, you can also set the property `svn:auto-props` on the root folder. The property value is automatically inherited by all child items.

Whichever method you choose, you should note that auto-props are only applied to files at the time they are added to the working copy. Auto-props will never change the properties of files which are already versioned.

Če želite stoodstotno zagotoviti, da imajo datoteke ob dodajanju v skladišče nastavljene ustrezne lastnosti, namestite v skladišče akcijske skripte, ki se izvedejo pred objavo (pre-hook script).

Objavite lastnosti

Lastnosti v sistemu Subversion so pod nadzorom različic. Potem, ko spremenite ali dodate lastnosti, morate spremembe objaviti.

Spori pri lastnostih

Če pri objavi sprememb pride do spora, ker je nek drug uporabnik spremenil isto lastnost, Subversion ustvari datoteko `.prej`. Ko rešite spor, to datoteko izbrišite.

4.18.2. Projektne lastnosti TortoiseSVN

TortoiseSVN ima nekaj svojih posebnih lastnosti, ki se začnejo s `tsvn:`.

- `tsvn:logminsize` nastavi najmanjšo dolžino sporočila dnevniškega zapisa za objavo. Če vnesete krajši zapis, objava ni mogoča. Ta lastnost je uporabna, ker vas opozori, da morate pri vsaki objavi podati primerno opisno sporočilo. Če ta lastnost ni nastavljena ali je nastavljena na vrednost nič, sistem omogoča objavo brez sporočila.

  `tsvn:lockmsgminsize` nastavi najmanjšo dolžino sporočila pri zaklepu. Če vnesete krajši zapis, pridobitev zaklepa ni mogoča. Ta lastnost je uporabna, ker vas opozori, da morate pri pridobivanju zaklepa podati primerno opisno sporočilo. Če ta lastnost ni nastavljena ali je nastavljena na vrednost nič, sistem omogoča pridobivanje zaklepa brez vnosa sporočila.

- `tsvn:logwidthmarker` se uporablja pri projektih, ki zahtevajo, da je sporočilo dnevniškega zapisa oblikovano tako, da imajo vrstice določeno največjo dolžino (tipično je ta dolžina 80 znakov) pred novo vrstico. Nastavitev vrednosti, različne od nič, ima dve posledici v oknu za vnos dnevniškega zapisa: nariše mejo največje dovoljene dolžine vrstice in onemogoči oblivanje besedila, tako da vidite, katere vrstice so predolge. Opomba: ta zmožnost pravilno deluje le, če je pisava, uporabljena za sporočila dnevniških zapisov, konstantne širine.

- `tsvn:logtemplate` se uporablja pri projektih, kjer je oblika sporočil dnevniških zapisov predpisana. Ta lastnosti vsebuje besedilo v več vrsticah, ki se vstavi v okno za vnos sporočila dnevniškega zapisa ob pričetku objave. Zapis nato uredite in vnesete zahtevane informacije. Opomba: če hkrati s to lastnostjo uporabljate tudi lastnost `tsvn:logminsize`, nastavite najmanjšo dolžino dnevniškega zapisa na vrednost, ki je večja od dolžine predloge, sicer določanje najmanjše velikosti zapisa nima več zaščitne vloge.

  There are also action specific templates which you can use instead of `tsvn:logtemplate`. The action specific templates are used if set, but `tsvn:logtemplate` will be used if no action specific template is set.

  The action specific templates are:

  - `tsvn:logtemplatecommit` is used for all commits from a working copy.

  - `tsvn:logtemplatebranch` is used when you create a branch/tag, or when you copy files or folders directly in the repository browser.

  - `tsvn:logtemplateimport` is used for imports.
  - `tsvn:logtemplatedelete` is used when deleting items directly in the repository browser.
  - `tsvn:logtemplatemove` is used when renaming or moving items in the repository browser.
  - `tsvn:logtemplatemkdir` is used when creating directories in the repository browser.
  - `tsvn:logtemplatepropset` is used when modifying properties in the repository browser.
  - `tsvn:logtemplatelock` is used when getting a lock.

- Subversion allows you to set “autoprops” which will be applied to newly added or imported files, based on the file extension. This depends on every client having set appropriate autoprops in their Subversion configuration file. `tsvn:autoprops` can be set on folders and these will be merged with the user's local autoprops when importing or adding files. The format is the same as for Subversion autoprops, e.g. `*.sh = svn:eol-style=native;svn:executable` sets two properties on files with the `.sh` extension.

  Če pride do spora med krajevnimi samodejnimi lastnostmi in lastnostjo `tsvn:autoprops`, se upoštevajo nastavitve projekta, ker so sprecifične za ta projekt.

  As of Subversion 1.8, you should use the property `svn:auto-props` instead of `tsvn:autoprops` since this has the very same functionality but works with all svn clients and is not specific to TortoiseSVN.

- V oknu za objave lahko prilepite seznam spremenjenih datotek, vključno s stanjem vsake datoteke (dodano, spremenjeno...). Lastnost `tsvn:logfilelistenglish` določa, ali se stanje vstavlja v angleškem jeziku ali v jeziku, ki ga uporabljate v programu TortoiseSVN. Če te lastnosti ne nastavite, se upošteva privzeta vrednost `true`, kar pomeni, da so podatki o stanju prilepljeni v angleškem jeziku.

- TortoiseSVN can use a spell checker. On Windows 10, the spell checker of the OS is used. On earlier Windows versions, it can use spell checker modules which are also used by OpenOffice and Mozilla. If you have those installed this property will determine which spell checker to use, i.e. in which language the log messages for your project should be written. `tsvn:projectlanguage` sets the language module the spell checking engine should use when you enter a log message. You can find the values for your language on this page: *MSDN: Language Identifiers* [http://msdn2.microsoft.com/en-us/library/ms776260.aspx].

  Vrednost lahko vnesete v desetiški obliki ali v šestnajstiški obliki - v tem primeru ji dodajte predpono `0x`. Primer: za uporabo ameriške angleščine vnesite `0x0409` ali `1033`.

- Lastnost `tsvn:logsummary` se uporablja za izluščenje dela sporočila dnevniškega zapisa, ki se izpiše v dnevniku kot povzetek sporočila.

  Vrednost lastnosti `tsvn:logsummary` mora biti enovrstični regularni izraz, ki vsebuje eno skupino. Rezultat te skupine se uporabi za povzetek.

  Primer: `\[SUMMARY\]:\s+(.*)` najde vso besedilo, ki se nahaja za nizom “[SUMMARY]” in ga uporabi kot povzetek.

- The property `tsvn:logrevregex` defines a regular expression which matches references to revisions in a log message. This is used in the log dialog to turn such references into links which when clicked will either scroll to that revision (if the revision is already shown in the log dialog, or if it's available from the log cache) or open a new log dialog showing that revision.

  The regular expression must match the whole reference, not just the revision number. The revision number is extracted from the matched reference string automatically.

  If this property is not set, a default regular expression is used to link revision references.

- There are several properties available to configure client-side hook scripts. Each property is for one specific hook script type.

The available properties/hook-scripts are

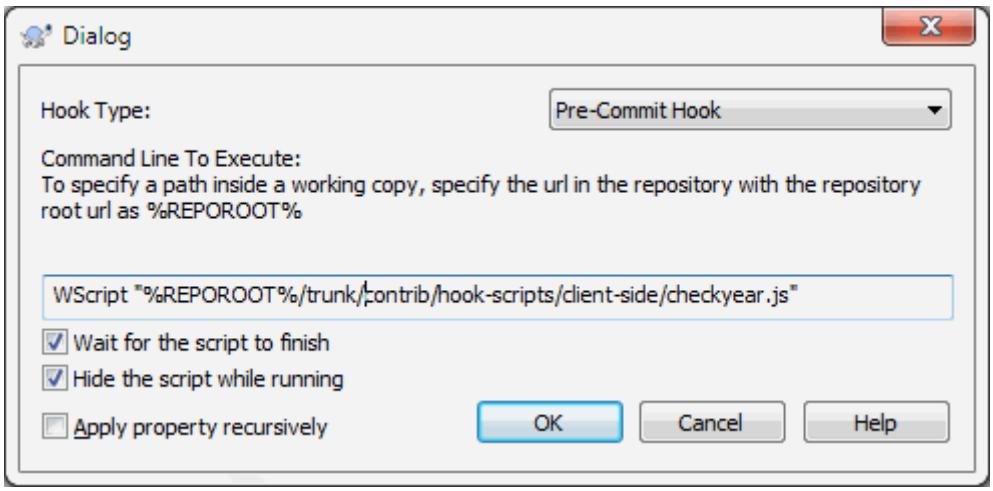
- tsvn:startcommithook
- tsvn:precommithook
- tsvn:postcommithook
- tsvn:startupdatehook
- tsvn:preupdatehook
- tsvn:postupdatehook
- tsvn:prelockhook
- tsvn:postlockhook

The parameters are the same as if you would configure the hook scripts in the settings dialog. See Razdelek 4.31.8, “Ukazne datoteke akcij na strani odjemalca” for the details.

Since not every user has his or her working copy checked out at the same location with the same name, you can configure a script/tool to execute that resides in your working copy by specifying the URL in the repository instead, using `%REPOROOT%` as the part of the URL to the repository root. For example, if your hook script is in your working copy under `contrib/hook-scripts/client-side/checkyear.js`, you would specify the path to the script as `%REPOROOT%/trunk/contrib/hook-scripts/client-side/checkyear.js`. This way even if you move your repository to another server you do not have to adjust the hook script properties.

Instead of `%REPOROOT%` you can also specify `%REPOROOT+%`. The + is used to insert any number of folder paths necessary to find the script. This is useful if you want to specify your script so that if you create a branch the script is still found even though the url of the working copy is now different. Using the example above, you would specify the path to the script as `%REPOROOT+%/contrib/hook-scripts/client-side/checkyear.js`.

The following screenshot shows how the script to check for current copyright years in source file headers is configured for TortoiseSVN.



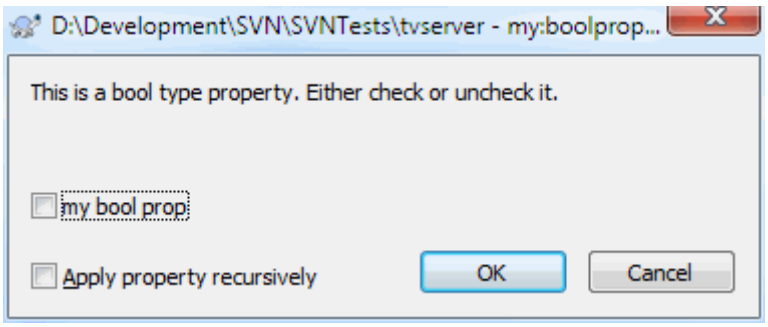
Slika 4.39. Property dialog for hook scripts

- Kadar želite dodati novo lastnost, jo lahko izberete iz seznama v spustnem polju ali pa vpišete poljubno ime lastnosti. Če na projektu uporabljate uporabniško določene lastnosti in želite, da se te lastnosti pojavijo v spustnem polju (v izogib tipkarskim napakam pri pisanju imena), lahko ustvarite seznam lastnih lastnosti z

uporabo `tsvn:userfileproperties` in `tsvn:userdirproperties`. Te lastnosti nastavite mapi. Ko boste urejali lastnosti kateregakoli elementa v tej mapi, se bodo lastne lastnosti pojavile na seznamu preddefiniranih imen.

You can also specify whether a custom dialog is used to add/edit your property. TortoiseSVN offers four different dialog, depending on the type of your property.

bool
: If your property can only have two states, e.g., true and false, then you can configure your property as a `bool` type.



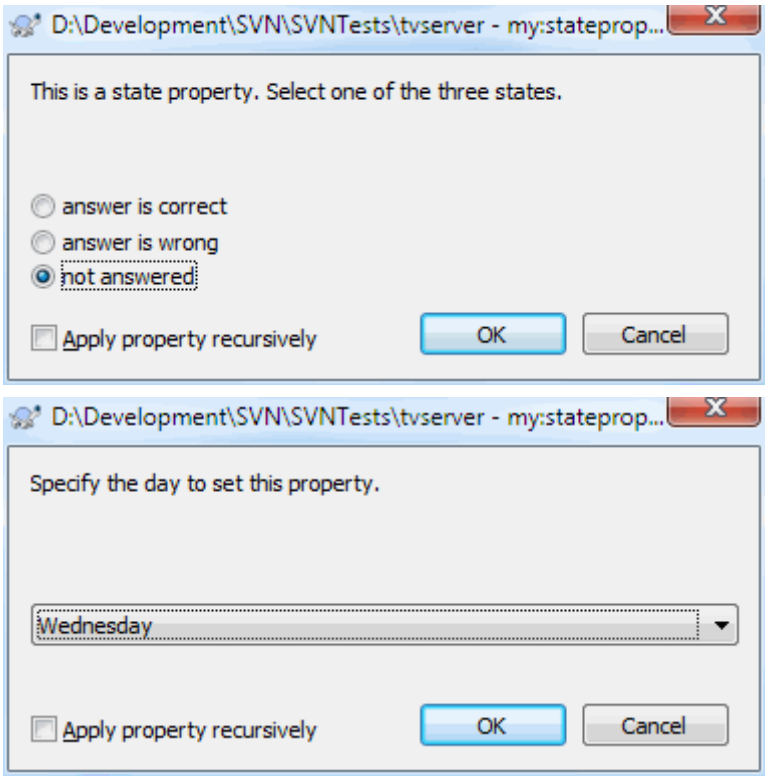
Slika 4.40. Property dialog boolean user types

Specify your property like this:

```
propertyname=bool;labeltext(YESVALUE;NOVALUE;Checkboxtext)
```

the `labeltext` is the text shown in the dialog above the checkbox where you can explain the purpose and use of the property. The other parameters should be self explanatory.

state
: If your property represents one of many possible states, e.g., `yes, no, maybe`, then you can configure your property as a `state`



Slika 4.41. Property dialog state user types

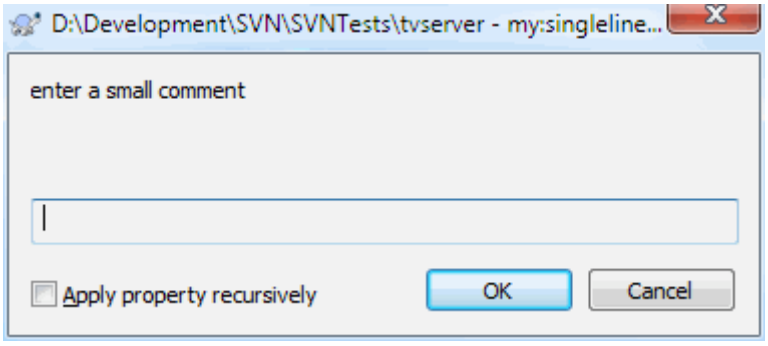
property like this:

```
propertyname=state;labeltext(DEFVAL;VAL1;TEXT1;VAL2;TEXT2;VAL3;TEXT3;...)
```

The parameters are the same as for the `bool` property, with `DEFVAL` being the default value to be used if the property isn't set yet or has a value that's not configured.

For up to three different values, the dialog shows up to three radio buttons. If there are more values configured, it uses a combo box from where the user can select the required state.

singleline
For properties that consist of one line of text, use the `singleline` property type:

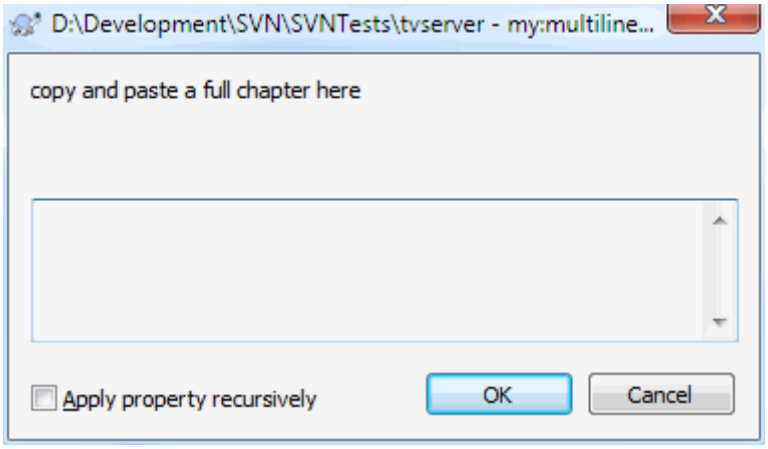


Slika 4.42. Property dialog single-line user types

```
propertyname=singleline;labeltext(regex)
```

the `regex` specifies a regular expression which is used to validate (match) the text the user entered. If the text does not match the regex, then the user is shown an error and the property isn't set.

multiline
: For properties that consist of multiple lines of text, use the `multiline` property type:



Slika 4.43. Property dialog multi-line user types

```
propertyname=multiline;labeltext(regex)
```

the `regex` specifies a regular expression which is used to validate (match) the text the user entered. Don't forget to include the newline (\n) character in the regex!

The screenshots above were made with the following `tsvn:userdirproperties`:

```
my:boolprop=bool;This is a bool type property. Either check or uncheck it.(true;false;
my:stateprop1=state;This is a state property. Select one of the two states.(true;true;
my:stateprop2=state;This is a state property. Select one of the three states.(maybe;tr
my:stateprop3=state;Specify the day to set this property.(1;1;Monday;2;Tuesday;3;Wedne
my:singlelineprop=singleline;enter a small comment(.*)
my:multilineprop=multiline;copy and paste a full chapter here(.*)
```

TortoiseSVN zna vključiti nekatera orodja za sledenje zadev. Za to se uporabljajo lastnosti, ki se začnejo z `bugtraq:`. Za več informacij preberite Razdelek 4.29, “Integracija s sistemi za sledenje zadev”.

Vključi lahko tudi nekatera spletna orodja za brskanje po skladiščih. Za to uporablja lastnosti, ki se začnejo z `webviewer:`. Za več informacij preberite Razdelek 4.30, “Integracija z internetno naravnanimi pregledovalniki skladišč”.

Nastavite projektne lastnosti na mapah

These special project properties must be set on *folders* for the system to work. When you use a TortoiseSVN command which uses these properties, the properties are read from the folder you clicked on. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (e.g. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it is sufficient to set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. If you set the same property but you use different values at different depths in your project hierarchy then you will get different results depending on where you click in the folder structure.

For project properties *only*, i.e. `tsvn:`, `bugtraq:` and `webviewer:` you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

When you add new sub-folders to a working copy using TortoiseSVN, any project properties present in the parent folder will automatically be added to the new child folder too.

Limitations Using the Repository Browser

Fetching properties remotely is a slow operation, so some of the features described above will not work in the repository browser as they do in a working copy.

- When you add a property using the repo browser, only the standard `svn:` properties are offered in the pre-defined list. Any other property name must be entered manually.
- Properties cannot be set or deleted recursively using the repo browser.
- Project properties will *not* be propagated automatically when a child folder is added using the repo browser.
- `tsvn:autoprops` will *not* set properties on files which are added using the repo browser.

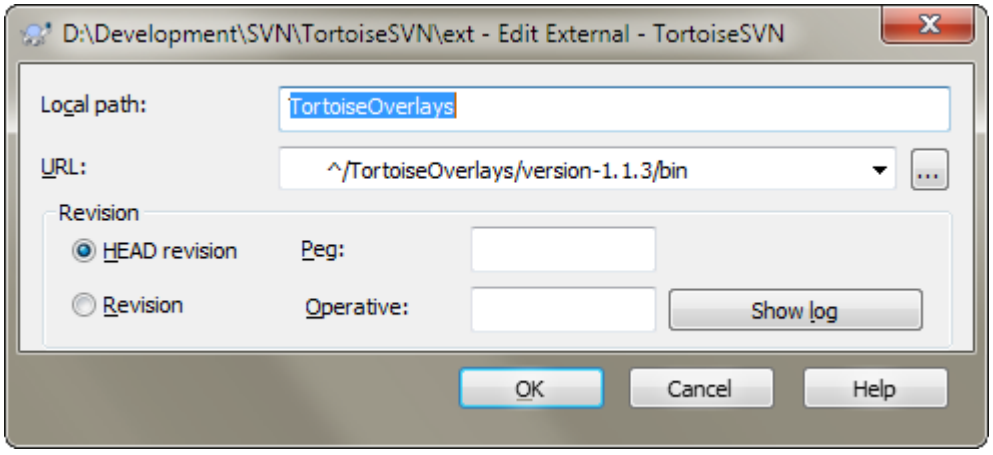
Opozorilo

Čeprav so projektne lastnosti TortoiseSVN zelo uporabne, delujejo le z odjemalcem TortoiseSVN, nekatere celo samo z novejšimi različicami TortoiseSVN. Če uporabniki na vašem projektu uporabljajo različne odjemalce za Subversion ali pa starejše različice TortoiseSVN, raje uporabite skripte akcij v skladišču za zagotavljanje spoštovanja pravil. Lastnosti `tsvn:` vam le pomagajo nastaviti pravila, ne morejo pa jih zagotavljati.

4.18.3. Property Editors

Some properties have to use specific values, or be formatted in a specific way in order to be used for automation. To help get the formatting correct, TortoiseSVN presents edit dialogs for some particular properties which show the possible values or break the property into its individual components.

4.18.3.1. External Content



Slika 4.44. svn:externals property page

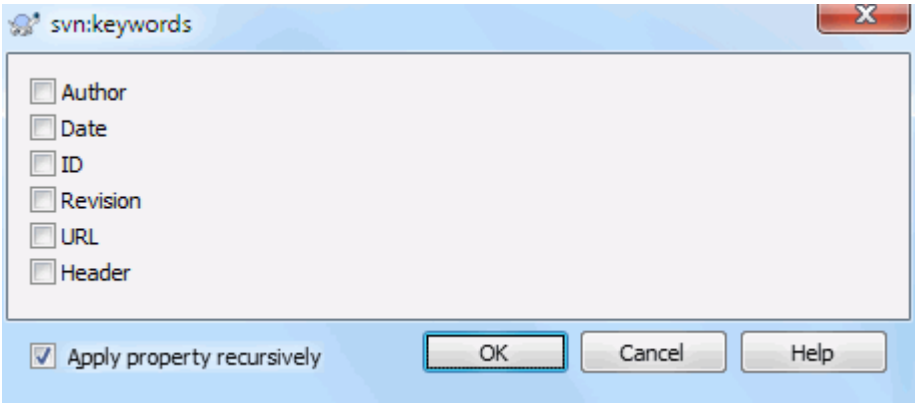
The `svn:externals` property can be used to pull in other projects from the same repository or a completely different repository as described in Razdelek 4.19, “External Items”.

You need to define the name of the sub-folder that the external folder is checked out as, and the Subversion URL of the external item. You can check out an external at its HEAD revision, so when the external item changes in the

repository, your working copy will receive those changes on update. However, if you want the external to reference a particular stable point then you can specify the specific revision to use. IN this case you may also want to specify the same revision as a peg revision. If the external item is renamed at some point in the future then Subversion will not be able to update this item in your working copy. By specifying a peg revision you tell Subversion to look for an item that had that name at the peg revision rather than at HEAD.

The button Find HEAD-Revision fetches the HEAD revision of every external URL and shows that HEAD revision in the rightmost column. After the HEAD revision is known, a simple right click on an external gives you the command to peg the selected externals to their explicit HEAD revision. In case the HEAD revision is not known yet, the right click command will fetch the HEAD revision first.

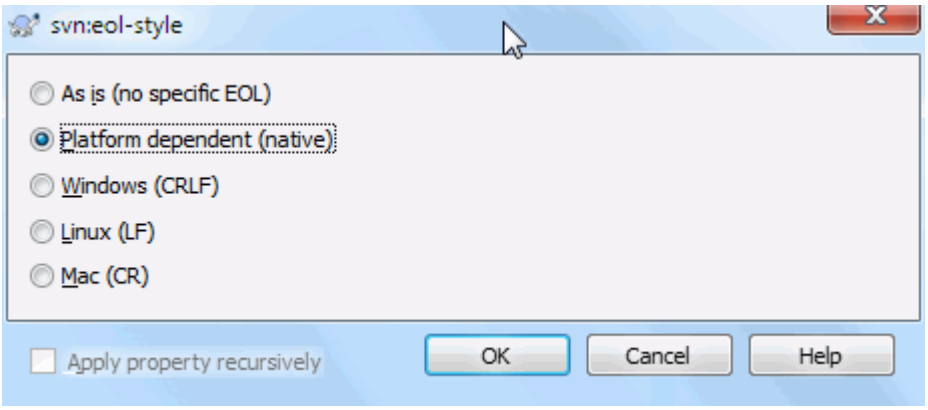
4.18.3.2. SVN Keywords



Slika 4.45. svn:keywords property page

Select the keywords that you would like to be expanded in your file.

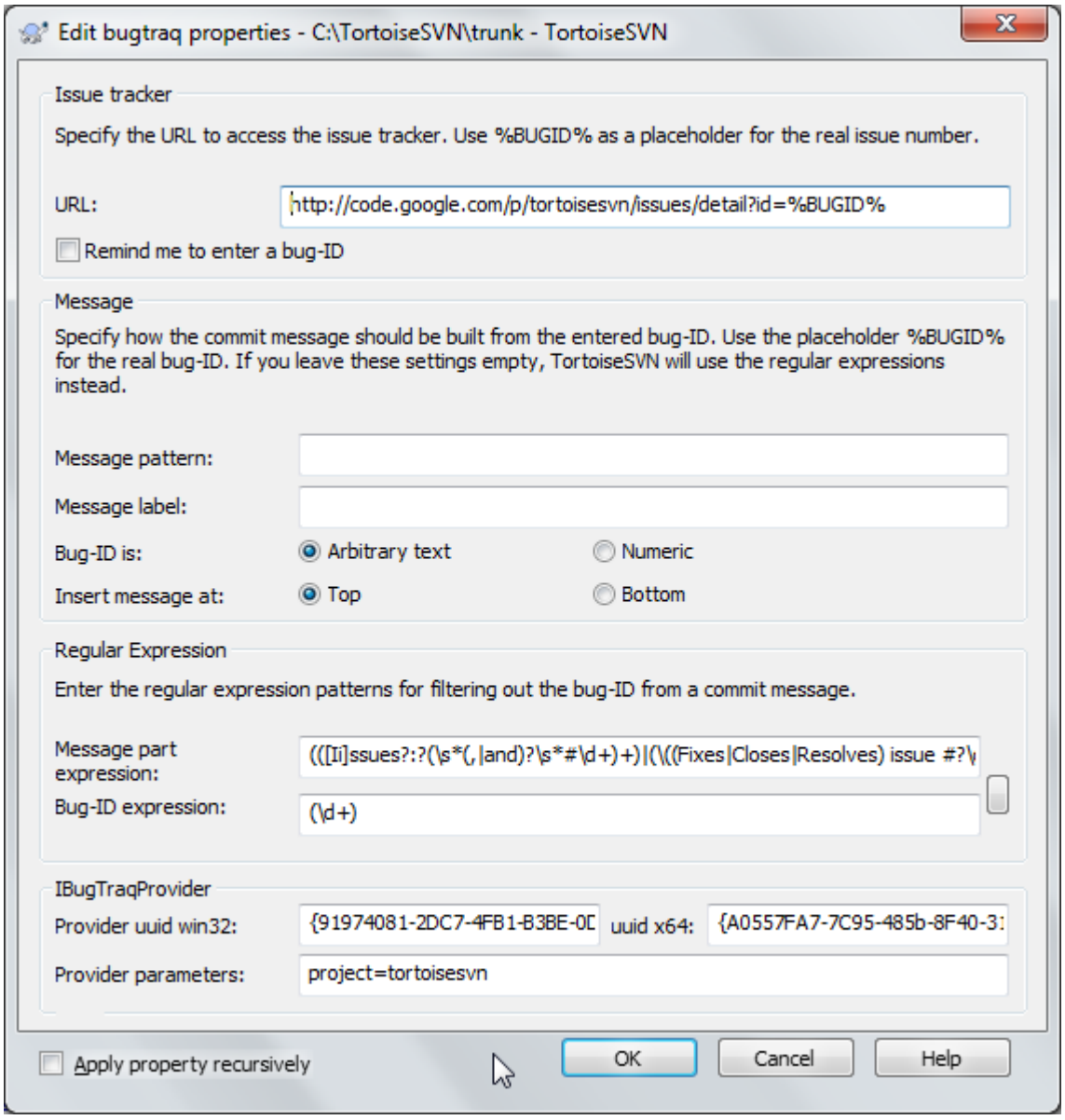
4.18.3.3. EOL Style



Slika 4.46. svn:eol-style property page

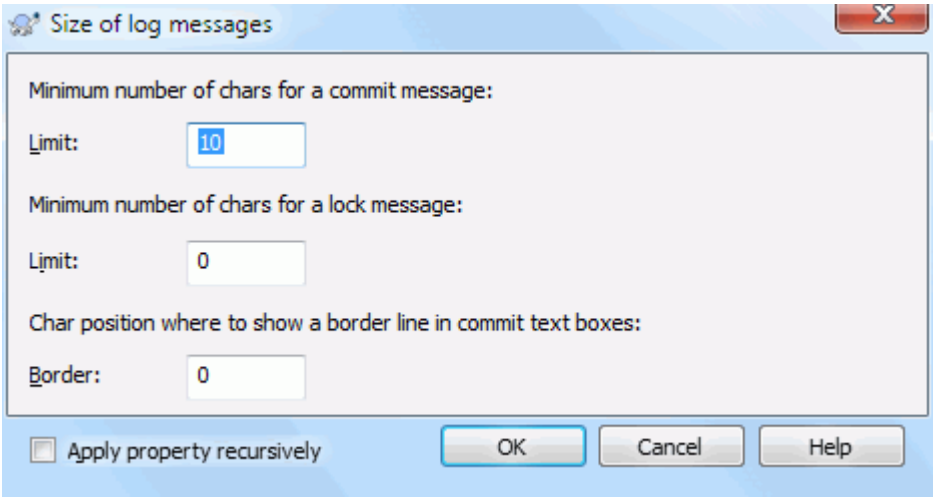
Select the end-of-line style that you wish to use and TortoiseSVN will use the correct property value.

4.18.3.4. Integracija sledilnika zadev



Slika 4.47. tsvn:bugtraq property page

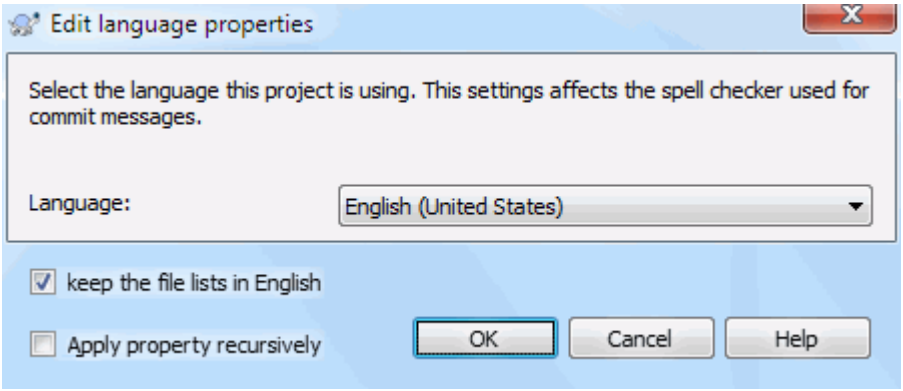
4.18.3.5. Log Message Sizes



Slika 4.48. Size of log messages property page

These 3 properties control the formatting of log messages. The first 2 disable the OK in the commit or lock dialogs until the message meets the minimum length. The border position shows a marker at the given column width as a guide for projects which have width limits on their log messages. Setting a value to zero will delete the property.

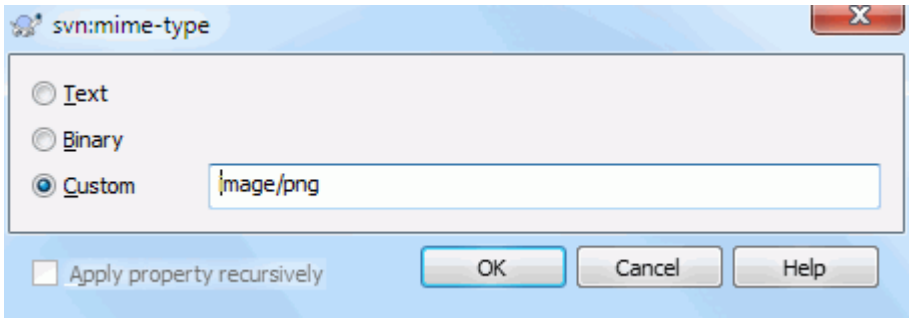
4.18.3.6. Project Language



Slika 4.49. Language property page

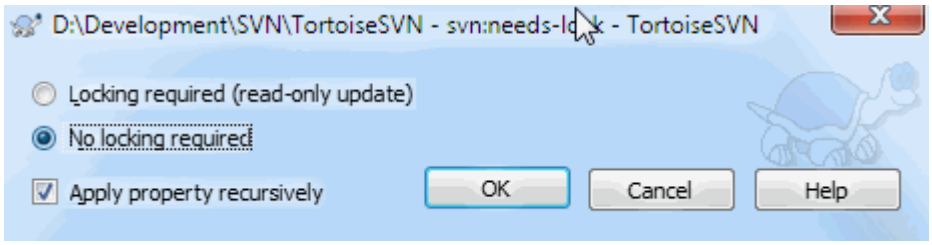
Choose the language to use for spell-checking log messages in the commit dialog. The file lists checkbox comes into effect when you right click in the log message pane and select Paste file list. By default the Subversion status will be shown in your local language. When this box is checked the status is always given in English, for projects which require English-only log messages.

4.18.3.7. MIME-type



Slika 4.50. svn:mime-type property page

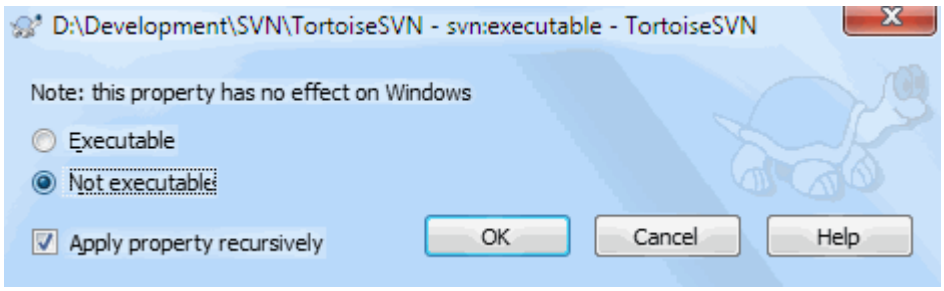
4.18.3.8. svn:needs-lock



Slika 4.51. svn:needs-lock property page

This property simply controls whether a file will be checked out as read-only if there is no lock held for it in the working copy.

4.18.3.9. svn:izvršljive-datoteke



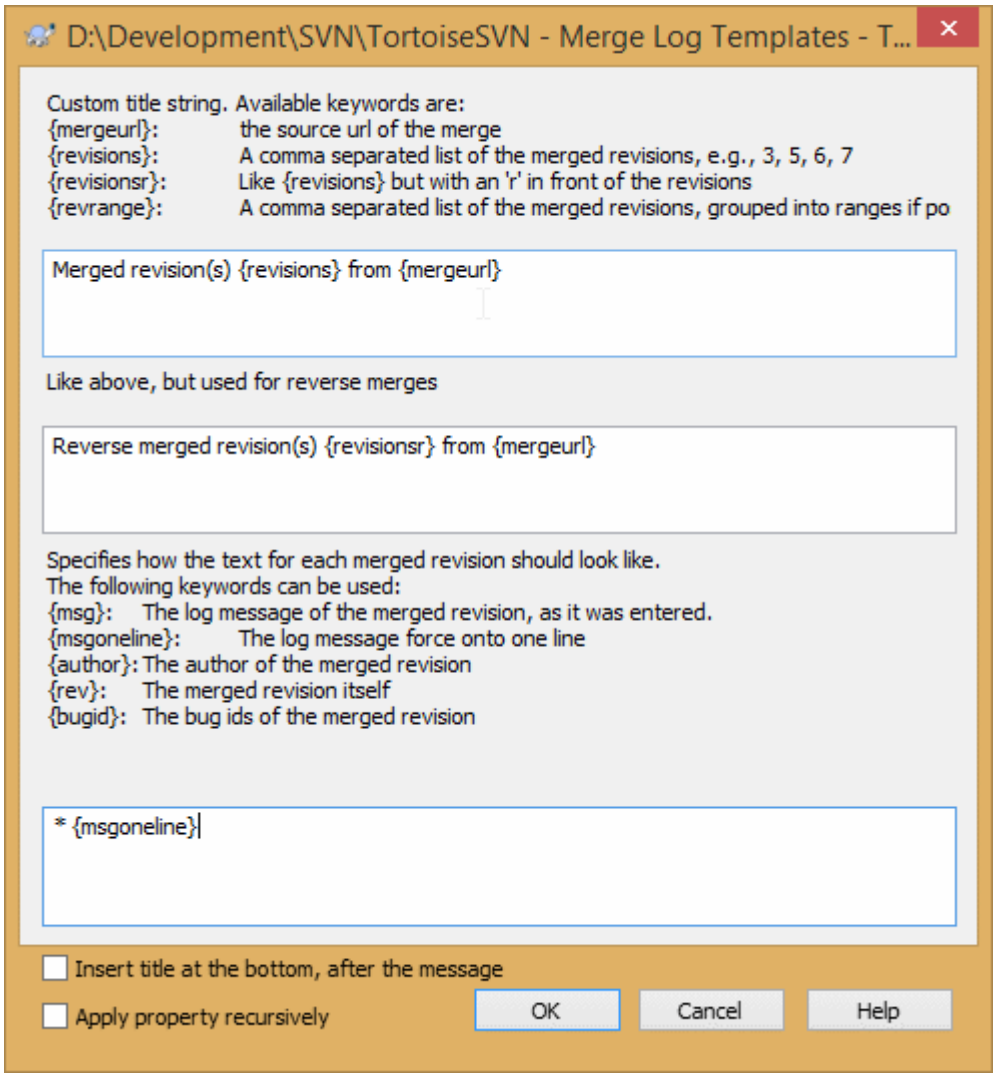
Slika 4.52. svn:executable property page

This property controls whether a file will be given executable status when checked out on a Unix/Linux system. It has no effect on a Windows checkout.

4.18.3.10. Merge log message templates

Whenever revisions are merged into a working copy, TortoiseSVN generates a log message from all the merged revisions. Those are then available from the Recent Messages button in the commit dialog.

You can customize that generated message with the following properties:



Slika 4.53. Property dialog merge log message templates

tsvn:mergelogtemplatetitle, tsvn:mergelogtemplatereversetitle
: This property specifies the first part of the generated log message. The following keywords can be used:

  {revisions}
  : A comma separated list of the merged revisions, e.g., `3, 5, 6, 7`

  {revisionsr}
  : Like `{revisions}`, but with each revision preceded with an `r`, e.g., `r3, r5, r6, r7`

  {revrange}
  : A comma separated list of the merged revisions, grouped into ranges if possible, e.g., `3, 5-7`

  {mergeurl}
  : The source URL of the merge, i.e., where the revisions are merged from.

  The default value for this string is `Merged revision(s) {revrange} from {mergeurl}:` with a newline at the end.

tsvn:mergelogtemplatemsg
: This property specifies how the text for each merged revision should look like. The following keywords can be used:

  {msg}
  : The log message of the merged revision, as it was entered.

{msgoneline}
Like `{msg}`, but all newlines are replaced with a space, so that the whole log message appears on one single line.

{author}
The author of the merged revision.

{rev}
The merged revision itself.

{bugids}
The bug IDs of the merged revision, if there are any.

tsvn:mergelogtemplatemsgtitlebottom
This property specifies the position of the title string specified with the `tsvn:mergelogtemplatetitle` or `tsvn:mergelogtemplatereversetitle`. If the property is set to `yes` or `true`, then the title string is appended at the bottom instead of the top.

Pomembno

This only works if the merged revisions are already in the log cache. If you have disabled the log cache or not shown the log first before the merge, the generated message won't contain any information about the merged revisions.

4.19. External Items

Sometimes it is useful to construct a working copy that is made out of a number of different checkouts. For example, you may want different files or subdirectories to come from different locations in a repository, or perhaps from different repositories altogether. If you want every user to have the same layout, you can define the `svn:externals` properties to pull in the specified resource at the locations where they are needed.

4.19.1. External Folders

Let's say you check out a working copy of `/project1` to `D:\dev\project1`. Select the folder `D:\dev\project1`, right click and choose Windows Menu → Properties from the context menu. The Properties Dialog comes up. Then go to the Subversion tab. There, you can set properties. Click Properties.... In the properties dialog, either double click on the `svn:externals` if it already exists, or click on the New... button and select `externals` from the menu. To add a new external, click the New... and then fill in the required information in the shown dialog.

Opozorilo

URLs must be properly escaped or they will not work, e.g. you must replace each space with `%20`.

If you want the local path to include spaces or other special characters, you can enclose it in double quotes, or you can use the \ (backslash) character as a Unix shell style escape character preceding any special character. Of course this also means that you must use / (forward slash) as a path delimiter. Note that this behaviour is new in Subversion 1.6 and will not work with older clients.

Use explicit revision numbers

You should strongly consider using explicit revision numbers in all of your externals definitions, as described above. Doing so means that you get to decide when to pull down a different snapshot of external information, and exactly which snapshot to pull. Besides the common sense aspect of not being surprised by changes to third-party repositories that you might not have any control over, using explicit revision numbers also means that as you backdate your working copy to a previous revision, your externals definitions will also revert to the way they looked in that previous revision, which

> in turn means that the external working copies will be updated to match the way *they* looked back when your repository was at that previous revision. For software projects, this could be the difference between a successful and a failed build of an older snapshot of your complex code base.
>
> The edit dialog for `svn:externals` properties allows you to select the externals and automatically set them explicitly to the HEAD revision.

If the external project is in the same repository, any changes you make there will be included in the commit list when you commit your main project.

If the external project is in a different repository, any changes you make to the external project will be shown or indicated when you commit the main project, but you have to commit those external changes separately.

Če uporabljate absolutne naslove URL v definicijah `svn:externals` in morate delovno kopijo premestiti (n. pr. če se spremeni naslov skladišča), se zunanji ne bodo spremenili in morda ne bodo več delovali.

V izogib tem težavam odjemalci Subversion verzije 1.5 ali novejše podpirajo relativne zunanje naslove URL. Na voljo so štirje načini podajanja relativnega naslova URL. V naslednjih primerih predpostavljamo, da imamo dve skladišči: enega na naslovu `http://primer.com/svn/sklad-1` in drugega na naslovu `http://primer.com/svn/sklad-2`. Prevzem naslova `http://primer.com/svn/sklad-1/projekt/trunk` naredimo v mapo `C:\Delo`, lastnost `svn:externals` pa je nastavljena na glavno vejo (trunk).

Relativno na nadrejeno mapo
: These URLs always begin with the string `../` for example:

```
../../widgets/foo  common/foo-widget
```

This will extract `http://example.com/svn/repos-1/widgets/foo` into `C:\Working\common\foo-widget`.

Upoštevajte, da je naslov URL relativen na naslov URL mape, ki ima nastavljeno lastnost `svn:externals` in ne na mapo, kjer se zunanji elementi nahajajo.

Relativno na korensko pot skladišča
: These URLs always begin with the string `^/` for example:

```
^/widgets/foo  common/foo-widget
```

This will extract `http://example.com/svn/repos-1/widgets/foo` into `C:\Working\common\foo-widget`.

You can easily refer to other repositories with the same `SVNParentPath` (a common directory holding several repositories). For example:

```
^/../repos-2/hammers/claw  common/claw-hammer
```

This will extract `http://example.com/svn/repos-2/hammers/claw` into `C:\Working\common\claw-hammer`.

Relativno na shemo
: URLs beginning with the string `//` copy only the scheme part of the URL. This is useful when the same hostname must the accessed with different schemes depending upon network location; e.g. clients in the intranet use `http://` while external clients use `svn+ssh://`. For example:

```
//example.com/svn/repos-1/widgets/foo  common/foo-widget
```

This will extract `http://example.com/svn/repos-1/widgets/foo` or `svn+ssh://example.com/svn/repos-1/widgets/foo` depending on which method was used to checkout `C:\Working`.

Relativno na naslov strežnika
: URLs beginning with the string `/` copy the scheme and the hostname part of the URL, for example:

```
/svn/repos-1/widgets/foo  common/foo-widget
```

This will extract `http://example.com/svn/repos-1/widgets/foo` into `C:\Working\common\foo-widget`. But if you checkout your working copy from another server at `svn+ssh://another.mirror.net/svn/repos-1/project1/trunk` then the external reference will extract `svn+ssh://another.mirror.net/svn/repos-1/widgets/foo`.

You can also specify a peg and operative revision for the URL if required. To learn more about peg and operative revisions, please read the *corresponding chapter* [http://svnbook.red-bean.com/en/1.8/svn.advanced.pegrevs.html] in the Subversion book.

Pomembno

If you specify the target folder for the external as a subfolder like in the examples above, make sure that *all* folders in between are versioned as well. So for the examples above, the folder `common` should be versioned!

While the external will work in most situations properly if folders in between are not versioned, there are some operations that won't work as you expect. And the status overlay icons in explorer will also not show the correct status.

Če želite izvedeti več o tem, kako Subversion obravnava lastnosti, preberite Razdelek 4.18, “Nastavitve projekta”.

Za pregled različnih načinov dostopa do skupnih projektov si preberite Razdelek B.6, “Kako vključim skupni podprojekt”.

4.19.2. External Files

As of Subversion 1.6 you can add single file externals to your working copy using the same syntax as for folders. However, there are some restrictions.

- The path to the file external must be a direct child of the folder where you set the `svn:externals` property.
- The URL for a file external must be in the same repository as the URL that the file external will be inserted into; inter-repository file externals are not supported.

A file external behaves just like any other versioned file in many respects, but they cannot be moved or deleted using the normal commands; the `svn:externals` property must be modified instead.

4.19.3. Creating externals via drag and drop

If you already have a working copy of the files or folders you want to include as externals in another working copy, you can simply add those via drag and drop from the windows explorer.

Simply right drag the file or folder from one working copy to where you want those to be included as externals. A context menu appears when you release the mouse button: SVN Add as externals here if you click on that context menu entry, the `svn:externals` property is automatically added. All you have to do after that is commit the property changes and update to get those externals properly included in your working copy.

4.20. Ustvarjanje vej/oznak

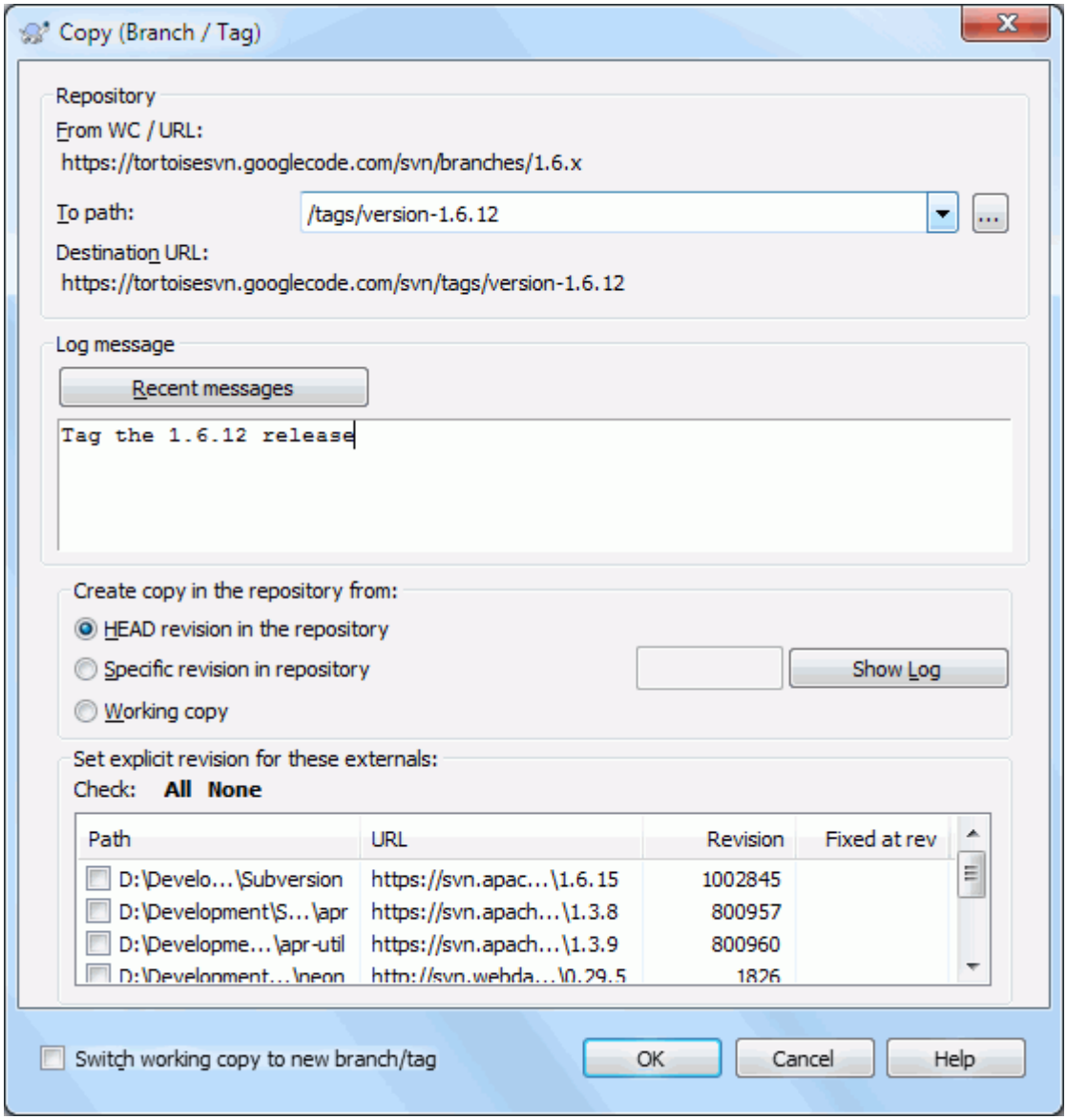
Ena od zmožnosti sistemov za nadzor različic je tudi možnost izoliranja sprememb v svojo razvojno linijo. Linija se prikaže kot *veja*. Veje se pogosto uporabljajo za preizkušanje novih stvari, da ne bi obremenjevali glavne razvojne linije z napakami prevajalnika in hrošči. Ko je nova zmožnost dovolj stabilna, se razvojna veja *spoji* nazaj v glavno vejo.

Še ena lastnost sistemov za nadzor različic je možnost označevanja določenih revizij (n. pr. verzije izdaje), tako da lahko kasneje ponovno zgradite izdajo ali nastavite razvojno okolje. Ta proces se imenuje *označevanje*.

Subversion nima posebnih ukazov za ustvarjanje vej ali oznak, ampak uporablja takoimenovane “poceni kopije”. Poceni kopije so podobne povezavam (hard links) v sistemu Unix, kar pomeni, da se namesto ustvarjanja kopije skladišča naredi notranja povezava, ki kaže na določeno drevo/revizijo. Zato je ustvarjanje vej in oznak zelo hitro, hkrati pa ne zavzema skoraj nobenega dodatnega prostora v skladišču.

4.20.1. Ustvarjanje veje ali oznake

Če ste projekt uvozili v priporočeno sturkturo map, je ustvarjanje vej ali oznak zelo preprosto:



Slika 4.54. Okno za za ustvarjanje veje/oznake

Izberite mapo v delovni kopiji, ki jo želite prekopirati v vejo ali oznako, nato pa izberite TortoiseSVN → Veja/oznaka....

The default destination URL for the new branch will be the source URL on which your working copy is based. You will need to edit that URL to the new path for your branch/tag. So instead of

```
http://svn.collab.net/repos/ProjectName/trunk
```

you might now use something like

```
http://svn.collab.net/repos/ProjectName/tags/Release_1.10
```

If you can't remember the naming convention you used last time, click the button on the right to open the repository browser so you can view the existing repository structure.

intermediate folders

When you specify the target URL, all the folders up to the last one must already exist or you will get an error message. In the above example, the URL `http://svn.collab.net/repos/ProjectName/tags/` must exist to create the `Release_1.10` tag.

However if you want to create a branch/tag to an URL that has intermediate folders that don't exist yet you can check the option `Create intermediate folders` at the bottom of the dialog. If that option is activated, all intermediate folders are automatically created.

Note that this option is disabled by default to avoid typos. For example, if you typed the target URL as `http://svn.collab.net/repos/ProjectName/Tags/Release_1.10` instead of `http://svn.collab.net/repos/ProjectName/tags/Release_1.10`, you would get an error with the option disabled, but with the option enabled a folder `Tags` would be automatically created, and you would end up with a folder `Tags` and a folder `tags`.

Sedaj morate izbrati vir kopiranja. Imate tri možnosti:

Revizija HEAD v skladišču
: Nova veja se skopira neposredno v skladišču iz revizije HEAD. Prenos podatkov iz delovne kopije ni potreben, zato je ustvarjanje veje zelo hitro.

Določena revizija v skladišču
: Nove veja se skopira neposredno v skladišču, vendar se lahko odločite za starejšo revizijo. To je uporabno, če ste prejšnji teden, ko ste naredili novo izdajo, pozabili kreirati oznako. Če se ne spomnite številke revizije, kliknite na gumb na desni in prikazalo se bo okno z dnevnikom revizij, od koder lahko izberete številko revizije. Iz delovne kopije se v skladišče ne prenašajo podatki, tako da je veja ustvarjena zelo hitro.

Delovna kopija
: Nova veja je identična kopija krajevne delovne kopije. Če ste nekatere datoteke posodobili na starejšo revizijo, bodo te starejše revizije datotek prav tako skopirane na vejo. Seveda se lahko pri takšnem zahtevnem tipu oznake zgodi, da se podatki prenašajo iz delovne kopije v skladišče, če tam še ne obstajajo.

Če želite, da se vaša delovna kopija samodejno preklopi na novo ustvarjeno vejo, vklopite potrditveno polje Preklopi delovno kopijo na drugo vejo / oznako. Če želite to narediti, se najprej prepričajte, da vaša delovna kopija nima krajevnih sprememb. Če jih ima, potem bodo le-te spojene v delovno kopijo veje, ko naredite preklop.

If your working copy has other projects included with `svn:externals` properties, those externals will be listed at the bottom of the branch/tag dialog. For each external, the target path and the source URL is shown.

If you want to make sure that the new tag always is in a consistent state, check all the externals to have their revisions pinned. If you don't check the externals and those externals point to a HEAD revision which might change in the future, checking out the new tag will check out that HEAD revision of the external and your tag might not compile anymore. So it's always a good idea to set the externals to an explicit revision when creating a tag.

The externals are automatically pinned to either the current HEAD revision or the working copy BASE revision, depending on the source of the branch/tag:

Copy Source	Pinned Revision
Revizija HEAD v skladišču	external's repos HEAD revision
Določena revizija v skladišču	external's repos HEAD revision
Delovna kopija	external's WC BASE revision

Tabela 4.1. Pinned Revision

externals within externals

If a project that is included as an external has itself included externals, then those will not be tagged! Only externals that are direct children can be tagged.

Če želite objaviti novo kopijo v skladišču, pritisnite gumb V redu. Ne pozabiti na dnevniški zapis. Ne pozabite, da se kopija ustvari *znotraj skladišča*.

Upoštevajte: če ne potrdite izbire za preklop nove veje na delovno kopijo, ustvarjanje vej in oznak *nima* vpliva na delovno kopijo. Tudi če vejo ustvarite iz delovne kopije, se spremembe objavijo na novi veji in ne na glavni veji, tako da je vaša delovna kopija še vedno lahko označena kot spremenjena (glede na glavno vejo).

4.20.2. Other ways to create a branch or tag

You can also create a branch or tag without having a working copy. To do that, open the repository browser. You can there drag folders to a new location. You have to hold down the **Ctrl** key while you drag to create a copy, otherwise the folder gets moved, not copied.

You can also drag a folder with the right mouse button. Once you release the mouse button you can choose from the context menu whether you want the folder to be moved or copied. Of course to create a branch or tag you must copy the folder, not move it.

Yet another way is from the log dialog. You can show the log dialog for e.g. trunk, select a revision (either the HEAD revision at the very top or an earlier revision), right click and choose create branch/tag from revision....

4.20.3. Prevzeti ali preklopiti...

... to (zares) ni vprašanje. Medtem ko prevzem prevzame vse datoteke iz izbrane veje v delovno kopijo, TortoiseSVN → Preklopi... prenese le spremenjene podatke v delovno kopijo. Dobro za obremenitev omrežja in vaše potrpljenje :-)

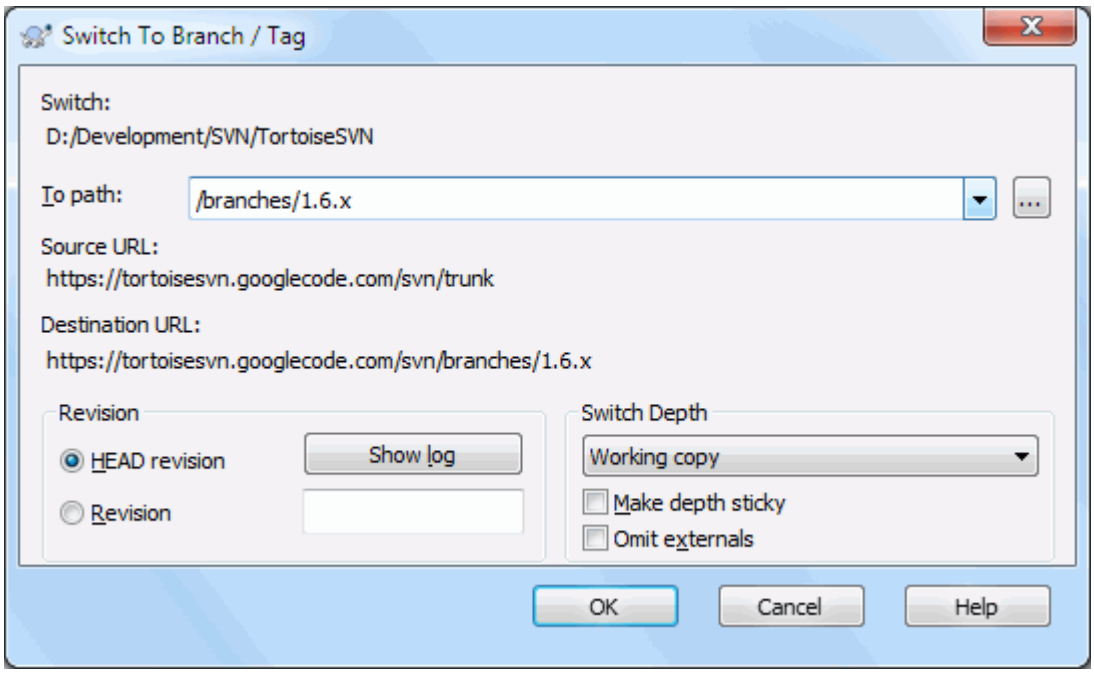
Na voljo je več načinov, kako začnete delati z novo ustvarjeno vejo ali oznako. Lahko:

- Uporabite ukaz TortoiseSVN → Prevzem in ustvarite svež prevzem v prazno mapo. Prevzem lahko naredite na katerokoli lokacijo na svojem disku. Naredite lahko poljubno število prevzemov iz skladišča.
- Preklopite delovno kopijo na novo ustvarjeno kopijo v skladišču. Izberite vrhnjo mapo projekta in iz kontekstnega menija uporabite TortoiseSVN → Preklopi...

  V naslednjem pogovornem oknu vnesite naslov URL veje, ki ste jo ustvarili. Izberite radijski gumb Revizija HEAD in kliknite na gumb V redu. S tem ste delovno kopijo preklopili na novo vejo/oznako.

  Preklop, enako kot posodobitev, nikoli ne povozi krajevnih sprememb. Vse spremembe, ki ste jih naredili v delovni kopiji in še niso bile objavljene, bodo spojene, ko naredite preklop. Če tega ne želite, potem morate spremembe objaviti ali pa delovno kopijio povrniti na objavljeno revizijo (tipično HEAD).

- Če želite delati na glavni in stranski veji, vendar ne želite delati svežih prevzemov, ki so časovno potratni, lahko naredite prevzem glavne veje, potem pa uporabite ukaz TortoiseSVN → Preklopi..., s katerim delovno kopijo posodobite na izbrano vejo.



Slika 4.55. Okno za preklop

Čeprav Subversion ne razlikuje med oznakami in vejami, se pri pri tipični uporabi ta dva ukaza nekoliko razlikujeta.

- Tags are typically used to create a static snapshot of the project at a particular stage. As such they are not normally used for development - that's what branches are for, which is the reason we recommended the `/trunk /branches /tags` repository structure in the first place. Working on a tag revision is *not a good idea*, but because your local files are not write protected there is nothing to stop you doing this by mistake. However, if you try to commit to a path in the repository which contains `/tags/`, TortoiseSVN will warn you.
- Včasih morate narediti dodatne spremembe na izdaji, ki ste jo že označili. Pravilen postopek je, da ustvarite novo vejo iz oznake in vejo objavite. Nato na novi veji naredite potrebne spremembe in ustvarite novo oznako iz veje, n. pr. `Verzija_1.0.1`.
- Če spremenite delovno kopijo, ustvarjeno iz veje, in jo objavite, gredo vse spremembe na novo vejo in *ne* na glavno vejo. Shranijo se le spremembe, nespremenjeni del pa ostane poceni kopija.

4.21. Spajanje

Kadar se veje uporabljajo za ločeno razvijanje, kasneje želite spremembe na eni veji spojiti nazaj v glavno vejo oz. obratno.

It is important to understand how branching and merging works in Subversion before you start using it, as it can become quite complex. It is highly recommended that you read the chapter *Branching and Merging* [http://svnbook.red-bean.com/en/1.8/svn.branchmerge.html] in the Subversion book, which gives a full description and many examples of how it is used.

Naslednja stvar, ki jo je treba poudariti, je, da se spajanje *vedno* izvaja v delovni kopiji. Če želite spremembe spojiti *v* vejo, morate imeti delovno kopijo, prevzeto iz te veje, in na njej izvesti ukaz TortoiseSVN → Spoji....

V splošnem je priporočljivo, da spajanje izvajate ob nespremenjeni delovni kopiji. Če ste naredili kakšne spremembe v delovni kopiji, jih najprej objavite. Če operacija spajanja ne bo uspela, boste verjetno želeli povrniti spremembe. Ukaz Povrni bo izbrisal *vse* spremembe na delovni kopiji, vključno s tistimi, ki ste jih naredili pred poskusom spajanja.

Poznamo tri načine spajanja, ki se med seboj nekoliko razlikujejo. Izbiro načina vam omogoča prva stran čarovnika za spajanje.

Spoji območje revizij
: Ta metoda pokriva primer, ko ste na veji (ali na glavni veji) naredili spremembe, ki jih sedaj želite prenesti na drugo vejo.

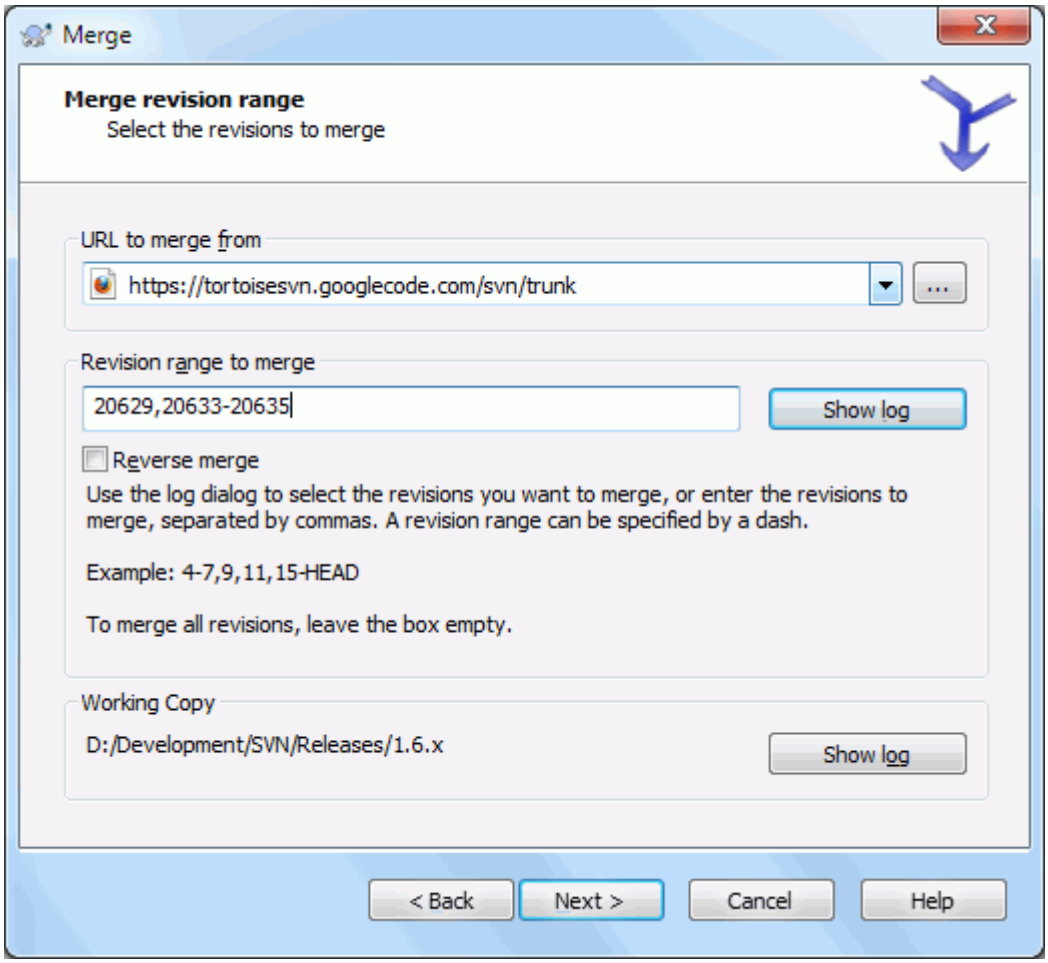
  What you are asking Subversion to do is this: “ Calculate the changes necessary to get [FROM] revision 1 of branch A [TO] revision 7 of branch A, and apply those changes to my working copy (of trunk or branch B). ”

  If you leave the revision range empty, Subversion uses the merge-tracking features to calculate the correct revision range to use. This is known as a reintegrate or automatic merge.

Spoji dveh različni drevesi
: This is a more general case of the reintegrate method. What you are asking Subversion to do is: “ Calculate the changes necessary to get [FROM] the head revision of the trunk [TO] the head revision of the branch, and apply those changes to my working copy (of the trunk). ” The net result is that trunk now looks exactly like the branch.

  If your server/repository does not support merge-tracking then this is the only way to merge a branch back to trunk. Another use case occurs when you are using vendor branches and you need to merge the changes following a new vendor drop into your trunk code. For more information read the chapter on *vendor branches* [http://svnbook.red-bean.com/en/1.8/svn.advanced.vendorbr.html] in the Subversion Book.

4.21.1. Spajanje območja revizij



Slika 4.56. Čarovnik za spajanje - Izberite obseg revizije

V polje Izvorni naslov URL za spajanje: vnesite poln naslov URL veje ali oznake s spremembami, ki jih želite spojiti v delovno kopijo. Lahko pa kliknete na gumb .... S tem pobrskate po skladišču in najdete željeno vejo. Če ste s te veje že spajali, lahko uporabite spustni seznam, ki prikazuje zgodovino že vnešenih naslovov URL.

If you are merging from a renamed or deleted branch then you will have to go back to a revision where that branch still existed. In this case you will also need to specify that revision as a peg revision in the range of revisions being merged (see below), otherwise the merge will fail when it can't find that path at HEAD.

V polje Območje revizij za spajanje vnesite seznam revizij, ki jih želite spojiti. Lahko vpišete eno revizijo, seznam določenih revizij, ločenih z vejicami, območje revizij, ločenih z vezajem ali kombinacijo vsega naštetega.

If you need to specify a peg revision for the merge, add the peg revision at the end of the revisions, e.g. `5-7,10@3`. In the above example, the revisions 5,6,7 and 10 would be merged, with 3 being the peg revision.

Pomembno

There is an important difference in the way a revision range is specified with TortoiseSVN compared to the command line client. The easiest way to visualise it is to think of a fence with posts and fence panels.

With the command line client you specify the changes to merge using two “fence post” revisions which specify the *before* and *after* points.

With TortoiseSVN you specify the changeset to merge using “fence panels”. The reason for this becomes clear when you use the log dialog to specify revisions to merge, where each revision appears as a changeset.

If you are merging revisions in chunks, the method shown in the Subversion book will have you merge 100-200 this time and 200-300 next time. With TortoiseSVN you would merge 100-200 this time and 201-300 next time.

This difference has generated a lot of heat on the mailing lists. We acknowledge that there is a difference from the command line client, but we believe that for the majority of GUI users it is easier to understand the method we have implemented.

Najlažji način, kako izberete večje število revizij, je s klikom na gumb Pokaži dnevnik; prikazalo se bo okno s seznamom zadnjih sprememb in pripadajočimi komentarji. Če želite spojiti spremembe iz ene revizije, izberite to revizijo. Če želite spojiti spremembe iz več revizij, izberite območje (z uporabo modifikatorja **Shift**). Kliknite na gumb V redu in seznam številk revizij se bo samodejno izpolnil.

Če želite spremembe spojiti nazaj *ven* iz delovne kopije (da prekličete že objavljeno spremembo), izberite revizije, ki jih želite povrniti in potrdite polje Obratno spajanje.

If you have already merged some changes from this branch, hopefully you will have made a note of the last revision merged in the log message when you committed the change. In that case, you can use Show Log for the Working Copy to trace that log message. Remembering that we are thinking of revisions as changesets, you should Use the revision after the end point of the last merge as the start point for this merge. For example, if you have merged revisions 37 to 39 last time, then the start point for this merge should be revision 40.

Če uporabljete sledenje spajanja, vam ni potrebno vedeti, katere revizije ste že spojili - Subversion si jih zapomni. Če pustite polje za vnos območja revizij prazno, bodo samodejno vključene vse revizije, ki še niso bile spojene. Za več informacij preberite Razdelek 4.21.5, “Sledenje spajanja”.

When merge tracking is used, the log dialog will show previously merged revisions, and revisions pre-dating the common ancestor point, i.e. before the branch was copied, as greyed out. The Hide non-mergeable revisions checkbox allows you to filter out these revisions completely so you see only the revisions which *can* be merged.

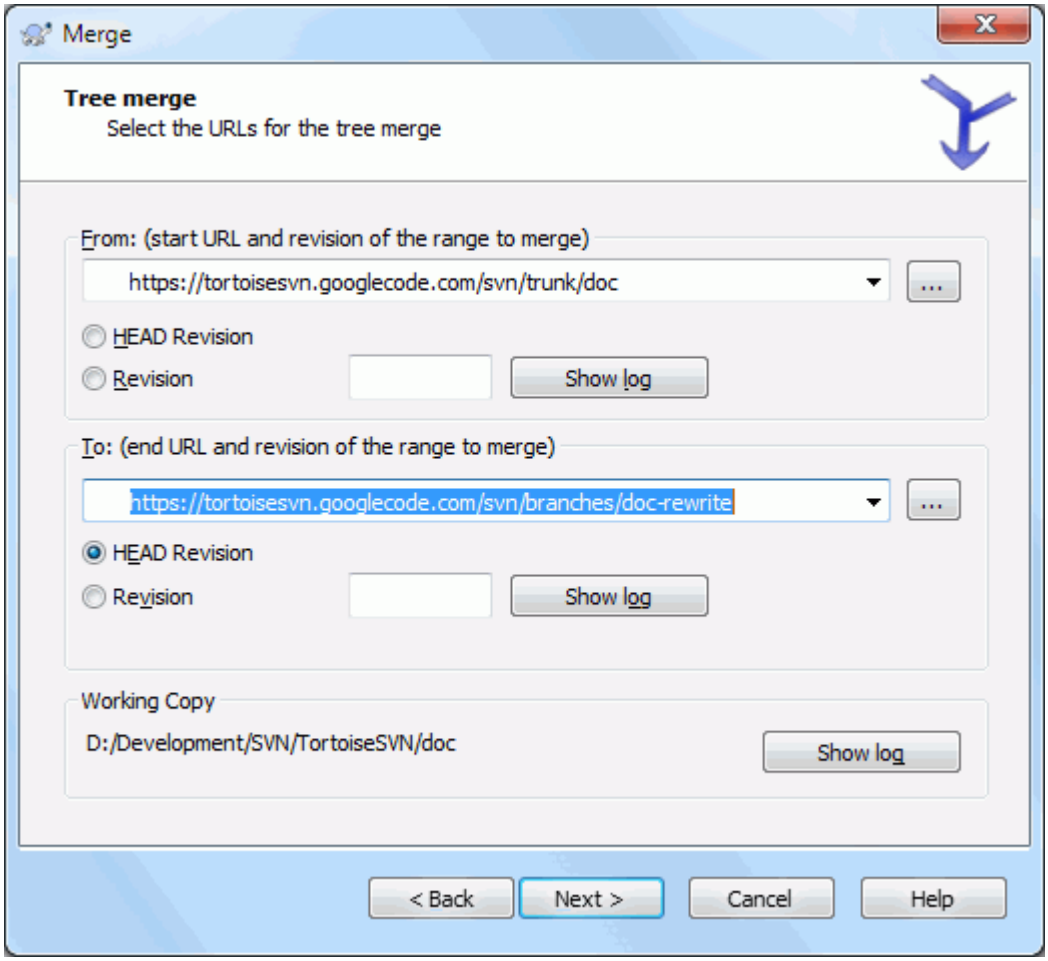
Če v skladišču objavljajo tudi drugi uporabniki, potem bodite pri uporabi spajanja do revizije HEAD previdni. Revizija HEAD mogoče ni tista, kot mislite, saj je od vaše zadnje posodobitve delovne kopije lahko kak drug uporabnik naredil objavo.

If you leave the range of revisions empty or have the radio button all revisions checked, then Subversion merges all not-yet merged revisions. This is known as a reintegrate or automatic merge.

Za izvajanje vključevanja veje morajo biti izpolnjeni določeni pogoji. Strežnik mora podpirati sledenje spajanja. Delovna kopija mora biti prevzeta v celoti (delni prevzemi niso dovoljeni) in biti brez krajevnih sprememb, preklopov ali elementov, posodobljenih na starejše revizije. Vse spremembe, narejene na glavni veji, morajo biti že spojene na stransko vejo (oziroma morajo biti označene kot spojene). Območje revizij za spajanje se določi samodejno.

Click Next and go to Razdelek 4.21.3, “Možnosti spajanja”.

4.21.2. Spajanje dveh različnih dreves



Slika 4.57. Čarovnik za spajanje - Spajanje dreves

Če ta način uporabljate za spajanje stranske veje na glavno vejo, morate čarovnika za spajanje zagnati iz delovnke kopije glavne veje.

V polje Od: vpišite poln naslov URL mape *glavne veje*. To se morda zdi napačno, a ne pozabite, da je glavna veja začetna točka spajanja, na katero želite dodati spremembe iz veje, kjer ste razvijali novo zmožnost. Uporabite lahko tudi gumb ... in pobrskate po skladišču.

V polje Do: vpišite poln naslov URL mape veje, kjer ste razvijali novo zmožnost.

V polji Revizija vpišite zadnji reviziji, pri kateri sta bili drevesi sihnronizirani. Če ste prepričani, da sprememb ne objavlja nihče drug, lahko v obeh primerih uporabite revizijo HEAD. Če obstaja možnost, da je po sinhronizaciji še kdo drug objavljal spremembe, potem raje uporabite točno določeno številko revizije, da se izognete izgubi sprememb iz novejših objav.

Za izbiro revizije lahko uporabite tudi gumb Pokaži dnevnik.

4.21.3. Možnosti spajanja

Ta stran čarovnika vam pred začetkom spajanja omogoča nastavitev naprednih možnosti. V večini primerov lahko uporabite kar privzete nastavitve.

Določite lahko globino spajanja, ki pove, kako globoko v delovno kopijo spajanje deluje. Za definicijo globine poglejte Razdelek 4.3.1, “Globina prevzema”. Privzeta globina je delovna kopija, ki uporablja obstoječo nastavitev globine, kar največkrat tudi potrebujete.

Most of the time you want merge to take account of the file's history, so that changes relative to a common ancestor are merged. Sometimes you may need to merge files which are perhaps related, but not in your repository. For example you may have imported versions 1 and 2 of a third party library into two separate directories. Although they are logically related, Subversion has no knowledge of this because it only sees the tarballs you imported. If you attempt to merge the difference between these two trees you would see a complete removal followed by a complete add. To make Subversion use only path-based differences rather than history-based differences, check the Ignore ancestry box. Read more about this topic in the Subversion book, *Noticing or Ignoring Ancestry* [http://svnbook.red-bean.com/en/1.8/svn.branchmerge.advanced.html#svn.branchmerge.advanced.ancestry].

Določite lahko, kako se upoštevajo zaključki vrstic in presledki. Možnosti so opisane v Razdelek 4.11.2, “Nastavitev zaključkov vrstic in presledkov”. Privzeto obnašanje je, da se zaključki vrstic in presledki obravnavajo enako kot vse druge spremembe.

The checkbox marked Force the merge is used to avoid a tree conflict where an incoming delete affects a file that is either modified locally or not versioned at all. If the file is deleted then there is no way to recover it, which is why that option is not checked by default.

Če uporabljate sledenje spajanja in želite revizijo označiti kot spojeno brez dejanskega spajanja, potrdite polje Zgolj zabeleži spajanje. To lahko naredite iz dveh razlogov. Lahko se zgodi, da je spajanje prezahtevno, da bi ga prepustili programu in naredite spremembe ročno, nato pa revizijo označite kot spojeno, da sistem sledenja spajanja ve za to. Drug razlog pa je, da želite preprečiti spajanje določene revizije. Če takšno revizijo označite kot spojeno, s tem preprečite ponovno spajanje (velja le za odjemalce, ki poznajo možnost sledenja spajanja).

Now everything is set up, all you have to do is click on the Merge button. If you want to preview the results Test Merge simulates the merge operation, but does *not* modify the working copy at all. It shows you a list of the files that will be changed by a real merge, and notes files where conflicts *may* occur. Because merge tracking makes the merge process a lot more complicated, there is no guaranteed way to find out in advance whether the merge will complete without conflicts, so files marked as conflicted in a test merge may in fact merge without any problem.

The merge progress dialog shows each stage of the merge, with the revision ranges involved. This may indicate one more revision than you were expecting. For example if you asked to merge revision 123 the progress dialog will report “ Merging revisions 122 through 123 ”. To understand this you need to remember that Merge is closely related to Diff. The merge process works by generating a list of differences between two points in the repository, and applying those differences to your working copy. The progress dialog is simply showing the start and end points for the diff.

4.21.4. Preverjanje rezultatov spajanja

Spajanje je sedaj zaključeno. Dobra praksa je, da rezultat spajanja preverimo. Spajanje je ponavadi kar komplicirano. Če je neka veja precej različna od glavne veje, se pri spajanju pogosto pojavijo spori.

Namig

Whenever revisions are merged into a working copy, TortoiseSVN generates a log message from all the merged revisions. Those are then available from the Recent Messages button in the commit dialog.

> To customize that generated message, set the corresponding project properties on your working copy. See Razdelek 4.18.3.10, “Merge log message templates”

Pri uporabi odjemalcev in strežnika Suversion pred različico 1.5 se informacije o spajanju ne shranjujejo in jih je zato treba voditi ločeno. Ko ste spajanje preverili in ste pripravljeni na objavo spojene revizije, v sporočilo dnevniškega zapisa *vedno* vpišite številke revizij, ki ste jih prenesli. Če boste kasneje spet spajali s te veje, boste morali vedeti, katere revizije ste že spojili. Sprememb ne smete spajati dvakrat. Za več informacij preberite *Best Practices for Merging* [http://svnbook.red-bean.com/en/1.4/svn.branchmerge.copychanges.html#svn.branchmerge.copychanges.bestprac] v knjigi The Subversion book.

Če strežnik in vsi odjemalci uporabljajo Subversion 1.5 ali novejši, zmožnost sledenje spajanja shrani spojene revizije in prepreči večkratno spajanje iste revizije. S tem si precej olajšate delo, saj lahko vedno izberete celotno območje revizij, spojene pa bodo le nove revizije.

Upravljanje vej je pomembno. Če želite biti na veji v koraku z glavno vejo, morate dovolj pogosto opravljati spajanje, da se veja in glavna veja ne oddaljita preveč ena od druge. Seveda se morate še vedno izogibati ponavljajočemu se spajanju sprememb, kot je opisano zgoraj.

Namig

V tem primeru veje za razvoj nove zmožnosti ne potrebujete več, ker ste novo kodo že integrirali na glavno vejo. Veja je sedaj odvečna in jo lahko izbrišete iz skladišča, če želite.

Pomembno

Subversion ne more spojiti datoteke z mapo in obratno - spaja lahko mape v mape in datoteke v datoteke. Če izberete datoteko in odprete okno za spajanje, potem morate v oknu navesti pot do datoteke. Če izberete mapo in odprete okno, potem morate vnesti naslov URL mape za spajanje.

4.21.5. Sledenje spajanja

Subversion 1.5 prinaša zmožnost sledenja spajanja. Ko spajate spremembe iz enega drevesa v drugega, se številke spojenih revizij shranijo. Te informacije se lahko nato uporabijo za različne namene.

- Večkratnemu spajanju iste revizije se lahko izognete. Ko je revizija označena kot spojena, jo bo Subversion pri naslednjih poskusil enostavno preskočil.
- Ko spremembe na veji spojite nazaj na glavno vejo, vam lahko dnevnik za glavno vejo prikaže tudi objave na posebni veji. S tem lažje sledite spremembam.
- When you show the log dialog from within the merge dialog, revisions already merged are shown in grey.
- Pri prikazu krivdnih informacij datoteke lahko prikažete avtorje spojenih revizij namesto avtorja, ki je opravil spajanje.
- Revizije lahko označite kot *ne spajaj*. To storite tako, da jih dodate na seznam spojenih revizij, čeprav se spajanje dejansko nikoli ni zgodilo.

Informacije o sledenju spajanja odjemalec ob spajanju shrani v lastnost `svn:mergeinfo`. Ko je spajanje objavljeno, strežnik te informacije shrani v bazo podatkov, zato se ob zahtevi po spajanju, prikazu dnevnika ali krivde lahko ustrezno odzove. Da bi sistem deloval, morate nadgraditi tako strežnik kot tudi vse odjemalce. Starejši odjemalci namreč ne shranjujejo lastnosti `svn:mergeinfo`, starejši strežniki pa ne pošiljajo zahtevanih informacij novim odjemalcem.

Find out more about merge tracking from Subversion's *Merge tracking documentation* [http://svn.apache.org/repos/asf/Subversion/trunk/notes/merge-tracking/index.html].

4.21.6. Handling Conflicts after Merge

Pomembno

The text in the conflict resolver dialogs are provided by the SVN library and might therefore not (yet) be translated as the TortoiseSVN dialogs are. Sorry for that.

Merging does not always go smoothly. Sometimes there is a conflict. TortoiseSVN helps you through this process by showing the *merge conflict* dialog.

Slika 4.58. The Merge Conflict Dialog

It is likely that some of the changes will have merged smoothly, while other local changes conflict with changes already committed to the repository. All changes which can be merged are merged. The Merge Conflict dialog gives you different ways of handling the lines which are in conflict.

For normal conflicts that happen due to changes in the file content or its properties, the dialog shows buttons which allow you to chose which of the conflicting parts to keep or reject.

Postpone
: Don't deal with the conflict now. Let the merge continue and resolve the conflicts after the merge is done.

Accept base
: This leaves the file as it was, without neither the changes coming from the merge nor the changes you've made in your working copy.

Accept incoming
: This discards all your local changes and uses the file as it arrives from the merge source.

Reject incoming
: This discards all the changes from the merge source and leaves the file with your local edits.

Accept incoming for conflicts
: This discards your local changes where they conflict with the changes from the merge source. But it leaves all your local changes which don't conflict.

Reject conflicts
: This discards changes from the merge source which conflict with your local changes. But it keeps all changes that don't conflict with your local changes.

Mark as resolved
: Marks the conflicts as resolved. This button is disabled until you use the button Edit to edit the conflict manually and save those changes back to the file. Once the changes are saved, the button becomes enabled.

Uredi
: Starts the merge editor so you can resolve the conflicts manually. Don't forget to save the file so the button Mark as resolved becomes enabled.

If there's a tree conflict, please first see Razdelek 4.6.3, “Tree Conflicts” about the various types of tree conflicts and how and why they can happen.

To resolve tree conflicts after a merge, a dialog is shown with various options on how to resolve the conflict:

Slika 4.59. The Merge Tree Conflict Dialog

Since there are various possible tree conflict situations, the dialog will show buttons to resolve those depending on the specific conflict. The button texts and labels explain what the option to resolve the conflict does. If you're not sure, either cancel the dialog or use the Postpone button to resolve the conflict later.

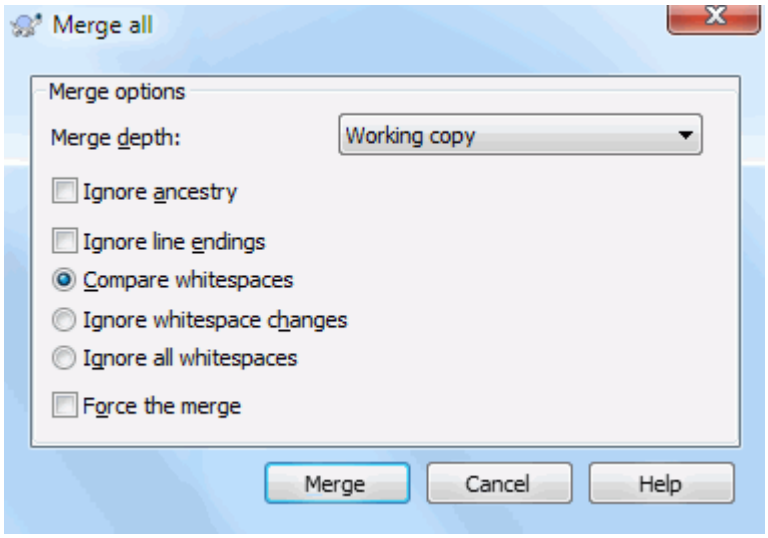
4.21.7. Vzdrževanje stranskih vej

Pri razvijanju nove zmožnosti na stranski veji se je koristno odločiti, kako se razvoj na stranski veji vključi v glavno vejo razvoja. Če se na glavni veji (`trunk`) dogajajo spremembe, se lahko zgodi, da se po določenem času veji zelo razlikujeta, tako da združevanje postane prava nočna mora.

Če je zmožnost enostavna, njen razvoj pa ne bo dolgotrajen, se lahko določite, da stransko vejo povsem ločite od glavne. Ko je razvoj na stranski veji končan, spremembe spojite nazaj na glavno vejo. V čarovniku za spajanje uporabite opcijo poji območje revizij, pri čemer za območje spajanja uporabite območje revizij stranske veje.

Če bo razvoj na stranski veji potekal dalj časa, pri tem pa morate upoštevati tudi spremembe na `glavni veji`, morate poskrbeti, da sta vejo sinhronizirani. To preprosto pomeni, da morate v določenih časovnih razmakih spajati spremembe iz glavne veje na stransko. Tako stranska veja vsebuje vse spremembe iz glavne veje, *poleg tega pa* tudi spremembe zaradi razvoja nove zmožnosti. Za sinhronizacij uporabljate možnost Spoji območje revizij. Ko je razvoj na stranski veji končan, lahko spojite spremembe nazaj na `glavno vejo` z uporabo možnosti Vključitev veje ali Spoji dve različni drevesi.

Another (fast) way to merge all changes from trunk to the feature branch is to use the TortoiseSVN → Merge all... from the extended context menu (hold down the **Shift** key while you right click on the file).



Slika 4.60. The Merge-All Dialog

This dialog is very easy. All you have to do is set the options for the merge, as described in Razdelek 4.21.3, “Možnosti spajanja”. The rest is done by TortoiseSVN automatically using merge tracking.

4.22. Zaklepanje

Subversion v splošnem najbolje deluje brez zaklepanja, z uporabo modela “kopiraj-spremeni-spoji”, kot je to opisano v Razdelek 2.2.3, “Rešitev kopiraj-spremeni-spoji”. Se pa vseeno pojavijo situacije, kjer je potrebno uveljaviti določeno obliko zaklepanja.

- Za delo uporabljate datoteke, ki jih ni mogoče spajati, naprimer grafične datoteke. Če dva uporabnika spremenita isto datoteko, spajanje ni mogoče, zato bo eden izmed njih svoje spremembe izgubil.
- Vaše podjetje je v preteklosti vedno uporabljalo nadzor različic z zaklepanjem, zato se je vodstvo odločilo, da je “zaklepanje najboljša rešitev”.

Najprej morate strežnik nadgraditi na različico Subversion 1.2 ali novejšo. Starejše različice namreč ne podpirajo zaklepanja. Če za dostop uporabljate protokol `file://`, potem morate seveda nadgraditi samo odjemalca.

The Three Meanings of “Lock”

In this section, and almost everywhere in this book, the words “lock” and “locking” describe a mechanism for mutual exclusion between users to avoid clashing commits. Unfortunately, there are two other sorts of “lock” with which Subversion, and therefore this book, sometimes needs to be concerned.

The second is `working copy locks`, used internally by Subversion to prevent clashes between multiple Subversion clients operating on the same working copy. Usually you get these locks whenever a command like update/commit/... is interrupted due to an error. These locks can be removed by running the cleanup command on the working copy, as described in Razdelek 4.17, “Čiščenje”.

And third, files and folders can get locked if they're in use by another process, for example if you have a word document opened in Word, that file is locked and can not be accessed by TortoiseSVN.

You can generally forget about these other kinds of locks until something goes wrong that requires you to care about them. In this book, “lock” means the first sort unless the contrary is either clear from context or explicitly stated.

4.22.1. Zaklepanje v sistemu Subversion

Po privzetih nastavitvah elementi niso zaklenjeni in vsakdo, ki ima pravice za pisanje v skladišču, lahko kadarkoli objavi spremembe katerekoli datoteke. Ostali uporabniki bodo periodično posodabljali svojo delovno kopijo. Spremembe v skladišču bodo spojene s krajevnimi spremembami.

Če za datoteko *pridobite zaklep*, potem lahko le vi objavite spremembe na tej datoteki. Objave s strani drugih uporabnikov bo sistem zaustavil, vse dokler zaklepa ne sprostite. Zaklenjene datoteke v skladišču ni mogoče spremeniti na noben način, tako da se je ne da izbrisati ali preimenovati, razen če niste ste lastnik zaklepa.

Pomembno

A lock is not assigned to a specific user, but to a specific user and a working copy. Having a lock in one working copy also prevents the same user from committing the locked file from another working copy.

As an example, imagine that user Jon has a working copy on his office PC. There he starts working on an image, and therefore acquires a lock on that file. When he leaves his office he's not finished yet with that file, so he doesn't release that lock. Back at home Jon also has a working copy and decides to work a little more on the project. But he can't modify or commit that same image file, because the lock for that file resides in his working copy in the office.

Ni pa nujno, da uporabniki vedo, da ste datoteko zaklenili. Če statusa zaklepanja ne preverjajo redno, bodo to opazili šele, ko bo puskusili datoteko objaviti in objava ne bo uspešna. To pa ni najbolj uporabno. Da bi bilo delo z zaklepi lažje, je na voljo nova lastnost `svn:needs-lock`. Če je ta lastnost nastavljena (na katerokoli vrednost), bo krajevna kopija datoteke ob prevzemu ali posodobitvi na voljo samo za branje, *razen* če je ta delovna kopija lastnica zaklepa za to datoteko. To deluje kot opozorilo, da datoteke ne urejajte, preden zanjo niste pridobili zaklepa. Datoteke pod nadzorom različic, ki so hkrati tudi označene samo za branje, imajo v programu TortoiseSVN posebno prekrivno ikono, ki vas opozarja, da morate pred urejanjem pridobiti zaklep.

Zaklep se pridobi za določeno delovno kopijo in lastnika. Če imate v lasti več delovnih kopij (doma, v službi), ste lahko lastnik zaklepa datoteke samo v *eni* od teh delovnih kopij.

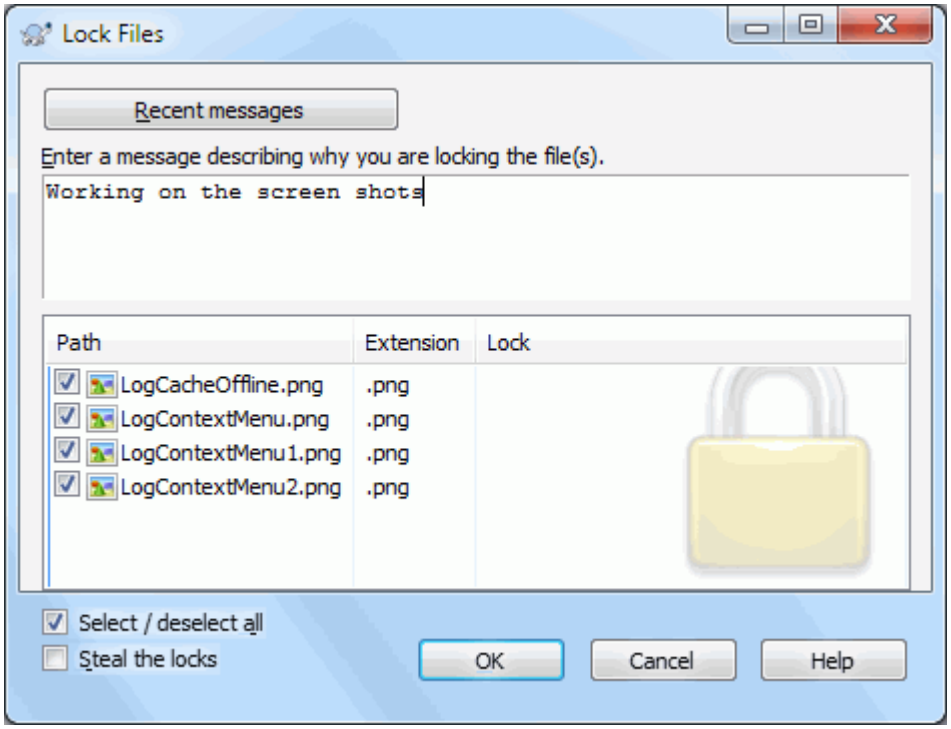
Če eden od vaših sodelavcev pridobi zaklep in odide na dopust, ne da bi zaklep sprostil, kaj storite? Subversion omogoča, da prisilno prevzamete zaklep. Sprostitev zaklepa nekoga drugega je znana pod imenom *prelom* zaklepa,

prisilen prevzem zaklepa, ki je v lasti nekoga drugega, pa *kraja* zaklepa. Seveda teh ukazov ne smete zlorabljati, če želite ostati v dobrih odnosih s sodelavci.

Zaklepi so shranjeni v skladišču, žeton za zaklep pa je shranjen v krajevni delovni kopiji. Če se zgodi prelom zaklepa, krajevni zaklep postane neveljaven. Informacija v skladišču se vedno uporablja kot edina referenčna.

4.22.2. Pridobivanje zaklepa

V delovni kopiji izberite datoteke, za katere želite pridobiti zaklep, potem pa izberite TortoiseSVN → Dobi zaklep....



Slika 4.61. Okno zaklepov

Pojavi se pogovorno okno, ki vam omogoča vpis sporočila, tako da lahko ostali uporabniki vidijo, zakaj ste datoteko zaklenili. Komentar ni obvezen in se uporablja le v primeru uporabe strežnika svnserve. Če (in *samo* če) morate nekomu ukrasti zaklep, potrdite polje Ukradi zaklep, potem pa kliknite na gumb V redu.

You can set the project property `tsvn:logtemplatelock` to provide a message template for users to fill in as the lock message. Refer to Razdelek 4.18, “Nastavitve projekta” for instructions on how to set properties.

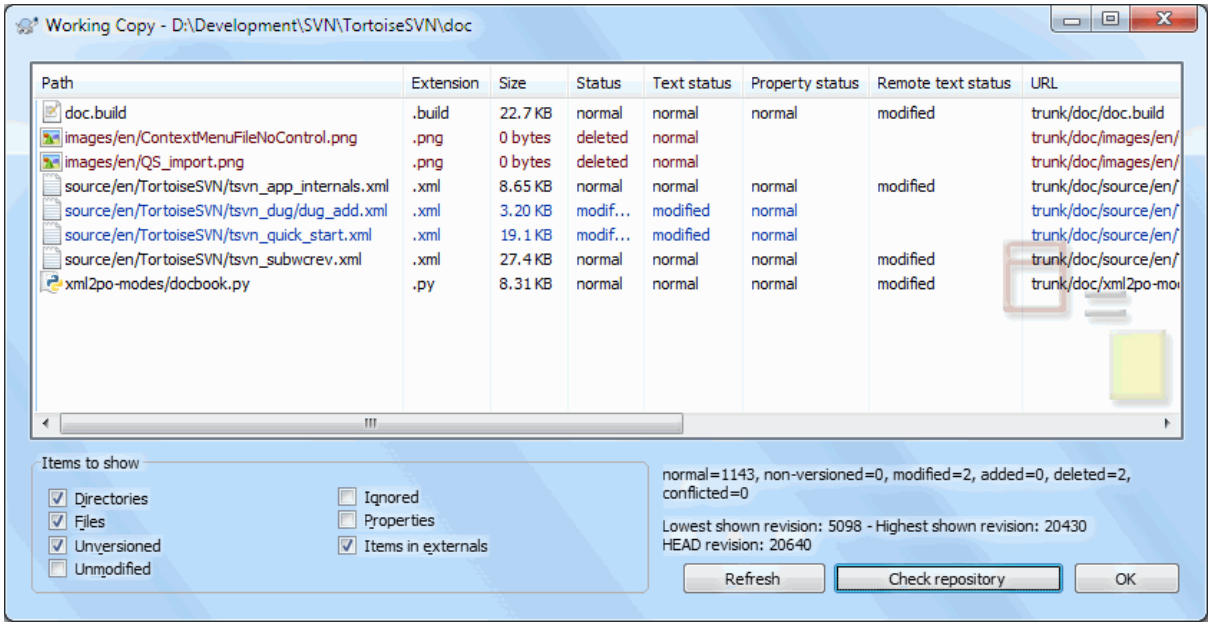
Če izberete mapo in potem uporabite TortoiseSVN → Dobi zaklep..., se bo pojavilo pogovorno okno Zaklep z *vsemi* datotekami v *vseh* podmapah. Če zares želite zakleniti celotni hierarhijo, je to pravi postopek, vendar lahko postanete zelo nepriljubljeni pri sodelavcih, če jim zaklenete celoten projekt. Bodite previdni...

4.22.3. Sprostitev zaklepa

Da ne bi pozabili sprostiti zaklepa, ki ga ne potrebujete več, so zaklenjene datoteke prikazane v oknu za objave in izbrane. Če nadaljujete z objavo, se zaklepi na izbranih datotekah sprostijo, tudi če datotek niste spreminjali. Če določenih zaklepov ne želite sprostiti, ustreznih datotek ne označite (če jih niste spreminjali). Če želite obdržati zaklepe na datotekah, ki ste jih spreminjali, pred objavo potrdite polje Obdrži zaklepe.

Za ročno sproščanje zaklepa v delovni kopiji izberite ustrezne datoteke in izvedite ukaz TortoiseSVN → Sprosti zaklep Ker dodatni podatki niso potrebni, TortoiseSVN kontaktira skladišče in sprosti zaklep. Ukaz lahko izvedete tudi na mapi. V tem primeru se sprostitev izvede rekurzivno na vseh zaklenjenih datotekah.

4.22.4. Preverjanje stanja zaklepanja



Slika 4.62. Pogovorno okno Preveri posodobitve

Za preverjanje, katere zaklepe imate v lasti vi in ostali uporabiniki, lahko uporabite ukaz TortoiseSVN → Preveri spremembe.... Zaklepi, ki so v vaši lasti, se pokažejo takoj. Za seznam zaklepov, ki jih imajo v lasti ostali uporabniki, uporabite gumb Preveri skladišče. S tem vidite tudi, če so kateri izmed vaših zaklepov prelomljeni ali ukradeni.

Iz kontekstnega menija lahko pridobite ali sprostite zaklep, prav tako pa lahko prelomite in ukradete zaklep, ki je v lasti drugih.

Izognite se prelamljanju in kraji zaklepov

Če nekomu prelomite ali ukradete zaklep, ne da bi uporabniku to povedali, lahko povzročite izgubo podatkov. Če delate z datotekami, ki se jih ne da spajati, in nekomu ukradete zaklep, lahko ta uporabnik po vaši sprostitvi zaklepa objavi svojo različico datoteke in s tem povozi vašo. Subversion nikoli ne izgubi podatkov, izgubite pa lahko varnost dela v teamu, ki vam jo daje zaklepanje.

4.22.5. Nastavite nezaklenjene datoteke samo za branje

Kot je že bilo omenjeno, je najbolj učinkovita metoda za uporabo zaklepanja uporaba lastnosti `svn:needs-lock`. Za več informacij o tem, kako datotekam nastaviti lastnosti, preberite Razdelek 4.18, “Nastavitve projekta”. Datoteke, ki imajo nastavljeno to lastnost, bodo ob prevzemu ali posodobitvi vedno dobile status datoteke samo za branje, razen če je delovna kopija lastnica zaklepa.

Kot opomnik, da je datoteka namenjena samo za branje, uporablja TortoiseSVN posebno prekrivno ikono.

Če velja na projektu pravilo, da je potrebno vsako datoteko zakleniti, potem bo verjetno lažje, da uporabite samodejne lastnosti (auto-props). Tako bo ta lastnost samodejno pripeta vsaki novo dodani datoteki. Za nadaljnje informacije preberite Razdelek 4.18.1.5, “Samodejna nastavitev lastnosti”.

4.22.6. Ukazna datoteka akcije za zaklepanje

Ko ustvarite novo skladišče s sistemom Subversion 1.2 ali novejšim, se v podmapi `hooks` ustvarijo štiri predloge za ukazne datoteke akcij. Kličejo se pred in po pridobivanju zaklepa ter pred in po sprostitvi zaklepa.

Na strežniku je priporočljivo nastaviti ukazne datoteke akcij za `post-lock` in `post-unlock`, ki pošljejo elektronsko pošto, ko se datoteka zaklene. S pomočjo teh datotek so vsi uporabniki obveščeni, ko nekdo zaklene/odklene datoteko. Primer ukazne datoteke se nahaja v datoteki `hooks/post-lock.tmpl` v mapi skladišča.

Nastacite lahko tudi ukazno datoteko akcije, ki preprečuje prelom ali krajo zaklepa, ali pa te operacije dovoli le administratorju. Lahko pa nastavite pošiljanje elektronske pošte lastniku zaklepa, kadar mu nekdo zaklep prelomi ali ukrade.

Za dodatne informacije preberite Razdelek 3.3, “Server side hook scripts”.

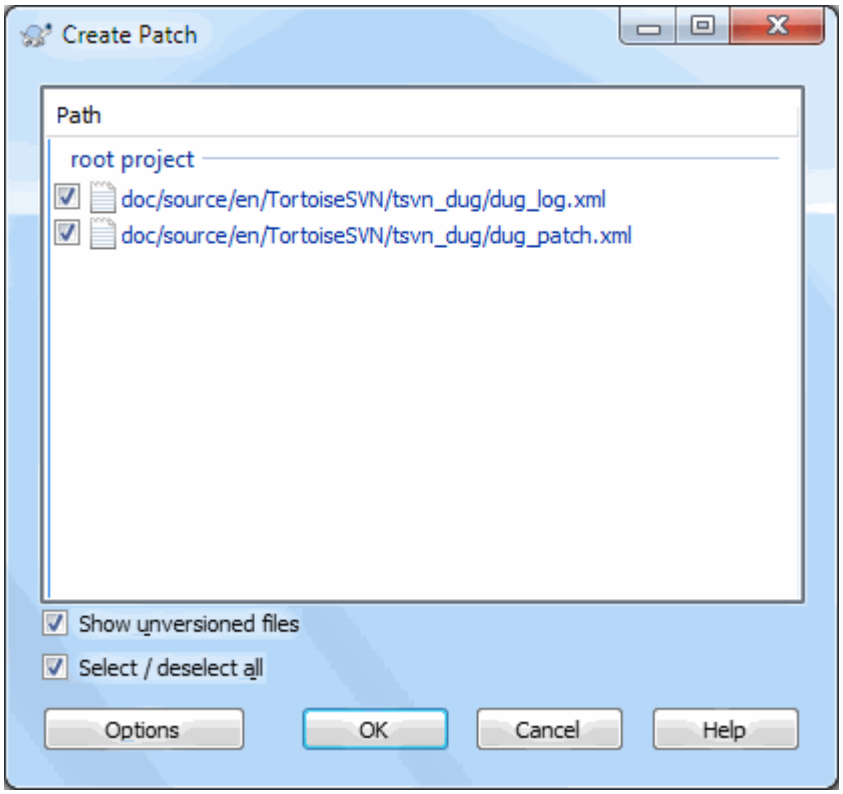
4.23. Ustvarjanje in nameščanje popravkov

V primeru odprtokodnih projektov (kot je tale) imajo pravice branja skladišča vsi in vsakdo lahko prispeva k projektu. Kako pa se ti prispevki nadzorujejo? Če bi lahko vsakdo objavljal spremembe, bi projekt kmalu postal nestabilen in neuporaben. V takšnem primeru se spremembe pošljejo razvojni ekipi, ki ima pravice za objavljanje, v obliki datoteke *popravkov*. Ekipa popravek pregleda in ga objavi v skladišču ali pa vrne avtorju.

Datoteke popravkov so poenotene razlike datotek, ki prikazujejo razlike med vašo delovno kopijo in osnovno revizijo.

4.23.1. Ustvarjanje datoteke popravkov

Najprej morate spremembe narediti *in jih preizkusiti*. Potem namesto ukaza TortoiseSVN → Objavi... na nadrejeni mapi uporabite TortoiseSVN → Ustvari popravek...



Slika 4.63. Okno za ustvarjanje popravkov

Sedaj izberete datoteke, ki jih želite vključiti v popravek, prav tako, kot to storite ob objavi. Ustvari se ena datoteka, ki vsebuje podatke o vseh spremembah, ki ste jih naredili na izbranih datotekah od zadnje posodobitve iz skladišča.

Stolpce v tem oknu lahko prilagodite enako kot stolpce v oknu Preveri spremembe. Za več informacij preberite Razdelek 4.7.3, “Krajevno in oddaljeno stanje”.

By clicking on the Options button you can specify how the patch is created. For example you can specify that changes in line endings or whitespaces are not included in the final patch file.

Izdelate lahko več popravkov, ki vsebujejo spremembe različnih skupin datotek. Seveda, če izdelate datoteko popravkov, nato naredite še dodatne spremembe in izdelate novo datoteko popravkov, bo ta vsebovala spremembe *obeh* skupin sprememb.

Shranite datoteko s poljubnim imenom. Datoteke popravkov imajo lahko poljubno končnico, vendar imajo po konvenciji končnico `.patch` ali `.diff`. Sedaj lahko datoteko oddate.

Namig

Do not save the patch file with a `.txt` extension if you intend to send it via email to someone else. Plain text files are often mangled with by the email software and it often happens that whitespaces and newline chars are automatically converted and compressed. If that happens, the patch won't apply smoothly. So use `.patch` or `.diff` as the extension when you save the patch file.

Popravek lahko shranite tudi v odložišče namesto v datoteko. Nato ga lahko prilepite v elektronsko sporočilo in ga pošljete v pregled ostalim. Če imate dve delovni kopiji na enem računalniku in želite prenesti spremembe iz ene kopije v drugo, je kopiranje popravka v odložišče pripraven način za izvedbo operacije.

If you prefer, you can create a patch file from within the Commit or Check for Modifications dialogs. Just select the files and use the context menu item to create a patch from those files. If you want to see the Options dialog you have to hold **shift** when you right click.

4.23.2. Nameščanje datoteke popravkov

Datoteke popravkov se namestijo v delovno kopijo. Nameščanje je potrebno narediti na istem nivoju map, na katerem je bil popravek narejen. Če ne veste, kateri nivo to je, poglejte v prvo vrstico datoteke popravkov. Primer: če je prva datoteka popravka `doc/source/english/chapter1.xml` in je prva vrstica popravka `Index: english/chapter1.xml`, potem morate popravek namestiti na mapo `doc/source/`. Vendar, če le delate na pravi delovni kopiji, bo TSVN zaznal, da ste izbrali napačen nivo in vam bo predlagal pravilnega.

Da bi namestili datoteko popravkov na delovni kopiji, morate imeti v skladišču vsaj pravice za branje. Razlog je v tem, da mora imeti program za spajanje referenco na revizijo, ki jo je oddaljeni programer uporabil kot osnovo.

From the context menu for that folder, click on TortoiseSVN → Apply Patch... This will bring up a file open dialog allowing you to select the patch file to apply. By default only `.patch` or `.diff` files are shown, but you can opt for “All files”. If you previously saved a patch to the clipboard, you can use Open from clipboard... in the file open dialog. Note that this option only appears if you saved the patch to the clipboard using TortoiseSVN → Create Patch.... Copying a patch to the clipboard from another app will not make the button appear.

Če ima datoteka popravkov končnico `.patch` ali `.diff`, lahko najno kliknete z desnim gumbom in izberete TortoiseSVN → Namesti popravek.... V tem primeru boste povprašani po lokaciji delovne kopije.

Ti dve metodi omogočata isti postopek na različna načina. S prvo metodo izberete delovno kopijo in poiščete datoteko popravkov, z drugo pa izberete datoteko popravkov in poiščete delovno kopijo.

Ko ste izbrali datoteko popravkov in lokacijo delovne kopije, TortoiseMerge začne postopek spajanja sprememb iz datoteke popravkov v delovno kopijo. Majhno okno vsebuje seznam datotek, ki so bile spremenjene. Dvokliknite na vsako izmed njih, preglejte spremembe in shranite spojene datoteke.

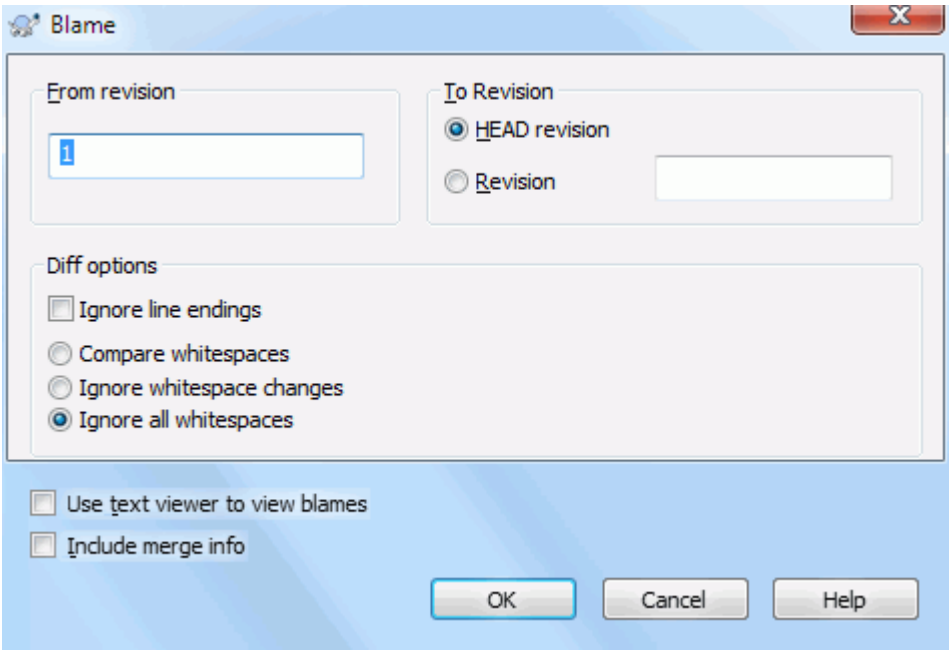
Popravek oddaljenega sodelavca je sedaj nameščen na vaši delovni kopiji, ki jo morate objaviti, da bodo lahko vsi uporabniki dostopali do sprememb v skladišču.

4.24. Kdo je spremenil posamezno vrstico?

Včasih vas ne zanima samo, katere vrstice so bile spremenjene, ampak tudi, kdo je določene vrstice spremenil. Takrat je uporaben ukaz TortoiseSVN → Okrivi.... Včasih ga imenujemo tudi *pohvali*.

Ta ukaz za vsako vrstico v datoteki izpiše avtorja in številko revizije, v kateri je bila vrstica spremenjena.

4.24.1. Okrivi datoteke



Slika 4.64. Okno hvali/okrivi

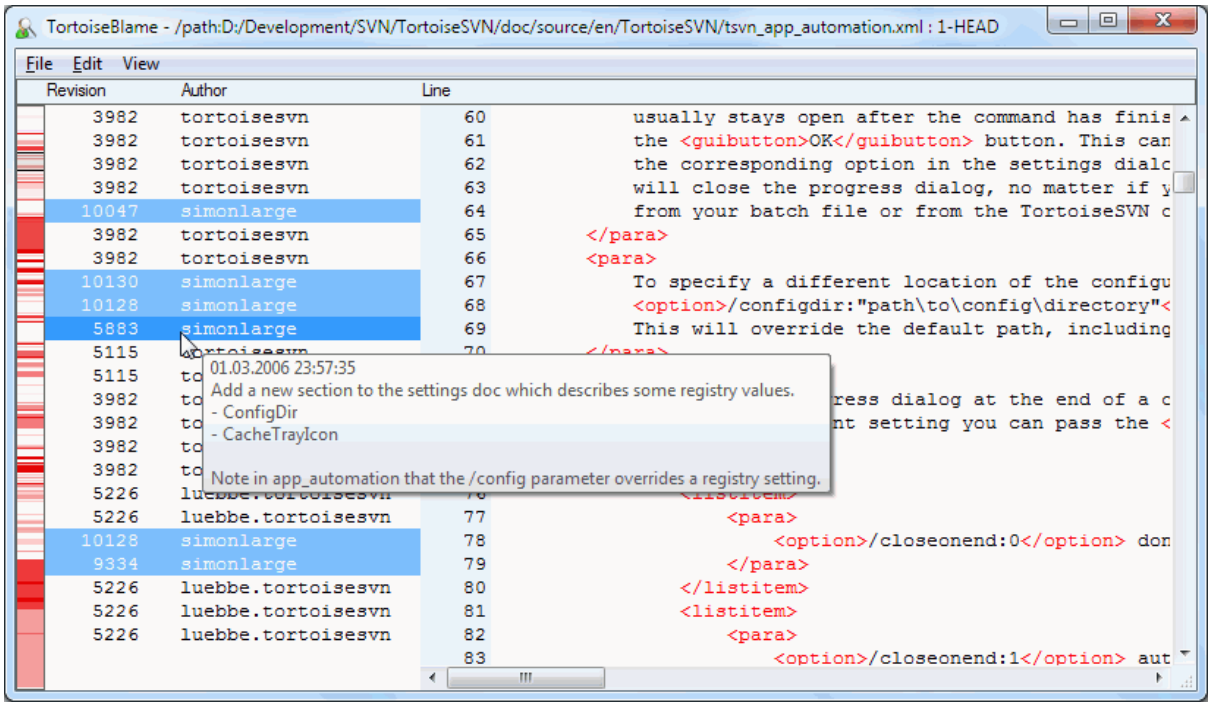
Če vas spremembe v zgodnejših revizijah ne zanimajo, lahko nastavite številko revizije, s katero naj se okrivljanje začne. Nastavite vrednost `1`, če želite okrivljanje *vseh* revizij.

By default the blame file is viewed using *TortoiseBlame*, which highlights the different revisions to make it easier to read. If you wish to print or edit the blame file, select Use Text viewer to view blames.

Nastavite lahko, kako se obravnavajo zaključki vrstic in presledki. Možnosti so opisane v Razdelek 4.11.2, “Nastavitev zaključkov vrstic in presledkov”. Privzeta nastavitev je, da se zaključki vrstic in presledki obravnavajo kot vse ostale spremembe. Če pa želite zamikanje prezreti in poiskati izvirnega avtorja, lahko tu nastavite ustrezno možnost.

You can include merge information as well if you wish, although this option can take considerably longer to retrieve from the server. When lines are merged from another source, the blame information shows the revision the change was made in the original source as well as the revision when it was merged into this file.

Once you press OK TortoiseSVN starts retrieving the data to create the blame file. Once the blame process has finished the result is written into a temporary file and you can view the results.



Slika 4.65. TortoiseBlame

TortoiseBlame, ki je del paketa TortoiseSVN, olajša pregledovanje datoteke krivd. Ko z miško nekaj trenutkov mirujete na vrstici v stolcu z informacijami, se vse vrstice iste revizije prikažejo s temnejšim ozadjem. Vrstice drugih revizij, ki jih je spremenil isti avtor, so prikazane s svetlim ozadjem. Barvno označevanje ne bo delovalo dobro, če uporabljate le 256 barv.

Če na vrstico kliknete z levim gumbom, se označijo vse vrstice iste revizije, vrstice, ki jih je spremenil isti avtor, pa so označene s svetlejšo barvo. Označevanje obvelja tudi če premaknete miško. Če želite označevanje izključiti, še enkrat kliknite na vrstico.

Komentar revizije (sporočilo dnevniškega zapisa) je prikazan v oknu z namigi, ko z miško za kratko počakate na stolpcu s krivdnimi informacijami. Če želite prekopirati dnevniško sporočilo za to revizijo, uporabite kontekstni meni, ki se pojavi ob desnem kliku.

Iskanje po krivdni datoteki izvedete z ukazom Uredi → Najdi.... Iščete lahko po številki revizije, avtorju in po sami vsebini datoteke. Sporočila dnevniških zapisov niso vključena v iskanje - za to uporabite Dnevnik.

Z ukazom Uredi → Pojdi na vrstico... skočite na določeno vrstico v datoteki.

Ko ste z miško nad stolcem s krivdnimi informacijami, lahko uporabite kontekstni meni, ki vam omogoča primerjanje revizij in pregled zgodovine. Številka revizije vrstice pod kazalcem miške se uporabi kot referenca. Ukaz Kontekstni meni → Okrivi prejšnjo revizijo ustvari krivdno poročilo za isto datoteko, vendar uporabi predhodno revizijo kot zgornjo mejo. To vam da krivdno poročilo za stanje datoteke točno preden se je vrstica, ki jo gledate, zadnjič spremenila. Ukaz Kontekstni meni → Pokaži spremembe požene pregledovalnik razlik, ki prikazuje, kaj se je spremenilo v referenčni reviziji. Ukaz Kontekstni meni → Prikaži dnevnik prikaže dnevnik, ki se začne z referenčno revizijo.

Če potrebujete boljše vizuelno razlikovanje med starimi in novimi spremembami, izberite View → Prikaz starosti vrstic z barvami. Novejše vrstice bodo prikazane z rdečimi barvnimi odtenki, starejše pa z modrimi. Privzete barve so dokaj svetle, vendar jih lahko spremenite v nastavitvah za TortoiseBlame.

If you are using Merge Tracking and you requested merge info when starting the blame, merged lines are shown slightly differently. Where a line has changed as a result of merging from another path, TortoiseBlame will show

the revision and author of the last change in the original file rather than the revision where the merge took place. These lines are indicated by showing the revision and author in italics. The revision where the merge took place is shown separately in the tooltip when you hover the mouse over the blame info columns. If you do not want merged lines shown in this way, uncheck the Include merge info checkbox when starting the blame.

If you want to see the paths involved in the merge, select View → Merge paths. This shows the path where the line was last changed, excluding changes resulting from a merge.

The revision shown in the blame information represents the last revision where the content of that line changed. If the file was created by copying another file, then until you change a line, its blame revision will show the last change in the original source file, not the revision where the copy was made. This also applies to the paths shown with merge info. The path shows the repository location where the last change was made to that line.

Nastavitve za TortoiseBlame prikažete z izbiro TortoiseSVN → Nastavitve... na zavihku TortoiseBlame. Preberite Razdelek 4.31.9, “Nastavitve TortoiseBlame”.

4.24.2. Okrivi spremembe

One of the limitations of the Blame report is that it only shows the file as it was in a particular revision, and the last person to change each line. Sometimes you want to know what change was made, as well as who made it. If you right click on a line in TortoiseBlame you have a context menu item to show the changes made in that revision. But if you want to see the changes *and* the blame information simultaneously then you need a combination of the diff and blame reports.

Dnevnik ponuja več možnosti, ki vam to omogočajo.

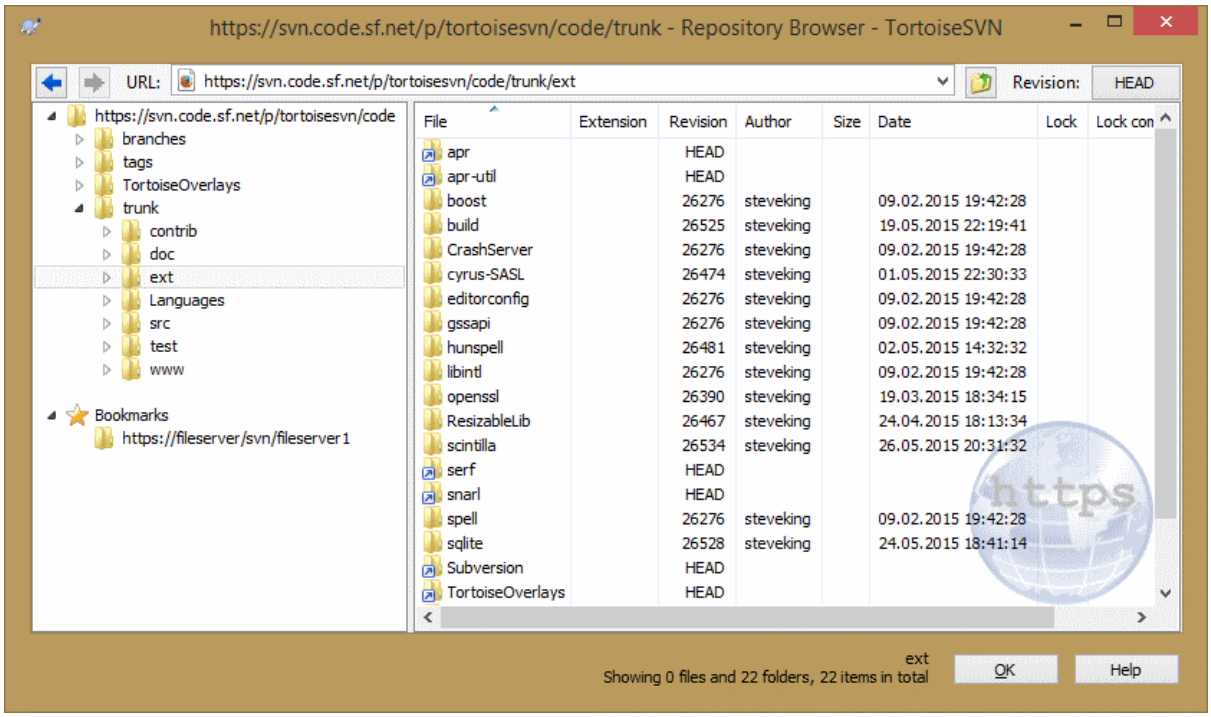
Okrivi revizije
: V zgornjem delu izberite dve reviziji in poženite Kontekstni meni → Okrivi revizije. S tem boste prenesli krivdne podatke za dve reviziji, potem pa vam bo pregledovalnik razlik prikazal razlike med obema krivdnima datotekama.

Okrivi spremembe
: V zgornjem delu izberite revizijo, potem pa v spodnjem delu izberite datoteko in izberite Kontekstni meni → Okrivi spremembe. S tem boste prenesli krivdne podatke za izbrano revizijo in prejšnjo revizijo, potem pa vam bo pregledovalnik razlik prikazal razlike med krivdnima datotekama.

Primerjaj in okrivi z delovno osnovo BASE
: Prikažite dnevnik za eno datoteko in v zgornjem delu izberite revizijo, potem pa poženite Kontekstni meni → Primerjaj in okrivi z delovno osnovo. S tem prenesete krivdne podatke za izbrano revizijo in za datoteko v delovni osnovi, potem pa se prikaže pregledovalnik razlik in premerja obe krivdni datoteki.

4.25. Brskalnik po skladišču

Včasih morate nekaj postoriti neposredno v skladišču, brez uporabe delovne kopije. Za to se uporablja *brskalnik po skladišču*. Prav tako kot vam Raziskovalec in prekrivne ikone omogočajo pregled delovne kopije, vam brskalnik po skladišču omogoča vpogled v strukturo in stanje skladišča.



Slika 4.66. Brskalnik po skladišču

Z brskalnikom po skladišču lahko izvajate kopiranje, premikanje, preimenovanje... neposredno v skladišču.

Brskalnik po skladišču je zelo podoben Raziskovalcu, s to razliko, da prikazuje vsebino skladišča v določeni reviziji namesto datotek na vašem računalniku. V levem delu lahko vidite strukturo map v obliki drevesa, v desnem pa vsebino izbrane mape. Na vrhu okna lahko vnesete naslov URL skladišča in revizijo, ki jo želite videti.

Folders included with the `svn:externals` property are also shown in the repository browser. Those folders are shown with a small arrow on them to indicate that they are not part of the repository structure, just links.

Tako kot v Raziskovalcu lahko tudi tu kliknete na glave stolpcev na desni strani, če želite definirati vrstni red razvrščanja. In prav tako kot v Raziskovalcu so v obeh delih na voljo kontekstni meniji.

Kontekstni meni za datoteko vam omogoča, da:

- Odprete izbrano datoteko s privzetim pregledovalnikom ali programom po lastni izbiri.
- Edit the selected file. This will checkout a temporary working copy and start the default editor for that file type. When you close the editor program, if changes were saved then a commit dialog appears, allowing you to enter a comment and commit the change.
- Prikažete dnevnik za datoteko ali graf vseh revizij, tako da lahko vidite, od kje datoteka prihaja
- Okrivite datoteko, da vidite, kdo in kdaj je spremenil določeno vrstico.
- Checkout a single file. This creates a “sparse” working copy which contains just this one file.
- Izbrišete ali preimenujete datoteko
- Shranite kopijo datoteke brez različic na trdi disk.
- Copy the URL shown in the address bar to the clipboard.
- Ustvarite kopijo datoteke - v drug del skladišča ali v delovno kopijo, ki kaže na isto skladišče.
- Pogledate/uredite lastnosti datoteke
- Create a shortcut so that you can quickly start repo browser again, opened directly at this location.

Kontekstni meni za mapo vam omogoča, da:

- Prikažete dnevnik za mapo ali graf vseh revizij, da vidite, od kje mapa prihaja.
- Izvozite mapo v mapo brez različic na trdem disku.
- Prevzamete mapo in si s tem ustvarite delovno kopijo na trdem disku.
- Ustvarite novo mapo v skladišču.
- Add unversioned files or folders directly to the repository. This is effectively the Subversion Import operation.
- Izbrišete ali preimenujete mapo.
- Make a copy of the folder, either to a different part of the repository, or to a working copy rooted in the same repository. This can also be used to create a branch/tag without the need to have a working copy checked out.
- Pogledate/uredite lastnosti mape
- Označite mapo za primerjanje. Označena mapa je izpisana polkrepko.
- Primerjate mapo s predhodno označeno mapo - kot poenoteno razliko ali kot seznam spremenjenih datotek, katerih razlike lahko vizuelno pregledate s privzetim orodjem za razlikovanje. To je uporabno pri primerjanju dveh oznak ali pri primerjanju veje in glavne veje, da vidite, kaj se je spremenilo.

Če na desni izberete dve mapi, lahko pregledate razlike med njima - kot poenoteno razliko ali kot seznam datotek, katerih razlike lahko vizuelno pregledate s privzetim orodjem za razlikovanje.

Če na desni strani izberete več map, jih lahko prevzamete sočasno v skupno mapo.

Če izberete dve oznaki, ki sta skopirani iz iste korenske mape (ponavadi `/trunk/`), lahko uporabite Kontekstni meni → Pokaži dnevnik... in s tem prikažete seznam revizij med obema oznakama.

External items (referenced using `svn:externals` are also shown in the repository browser, and you can even drill down into the folder contents. External items are marked with a red arrow over the item.

Kot je to običajno, lahko uporabite tipko **F5** za osvežitev pogleda. S tem osvežite vse, kar je trenutno prikazano. Če želite v naprej prenesti ali osvežiti informacije o vozliščih, ki še niso bila odprta, uporabite **Ctrl-F5**. Po tem bo razširjanje kateregakoli vozlišča potekalo zelo hitro, saj ne bo zamud zaradi prenosa podatkov preko omrežja.

Brskalnik po skladišču lahko uporabite tudi za izvajanje operacij z metodo povleci-in-spusti. Če povlečete v brskalnik po skladišču mapo iz Raziskovalca, jo s tem uvozite. Upoštevajte, da se v primeru, ko povlečete več elementov naenkrat, elementi uvozijo v ločenih objavah.

Če želite elemente premikati znotraj skladišča, jih enostavno povlecite z levim gumbom na novo lokacijo. Če želite namesto premikanja ustvariti kopijo, ob potegu držite pritisnjeno tipko **CTRL**. Pri kopiranju ima kazalnik dodan znak “plus”, prav tako kot v Raziskovalcu.

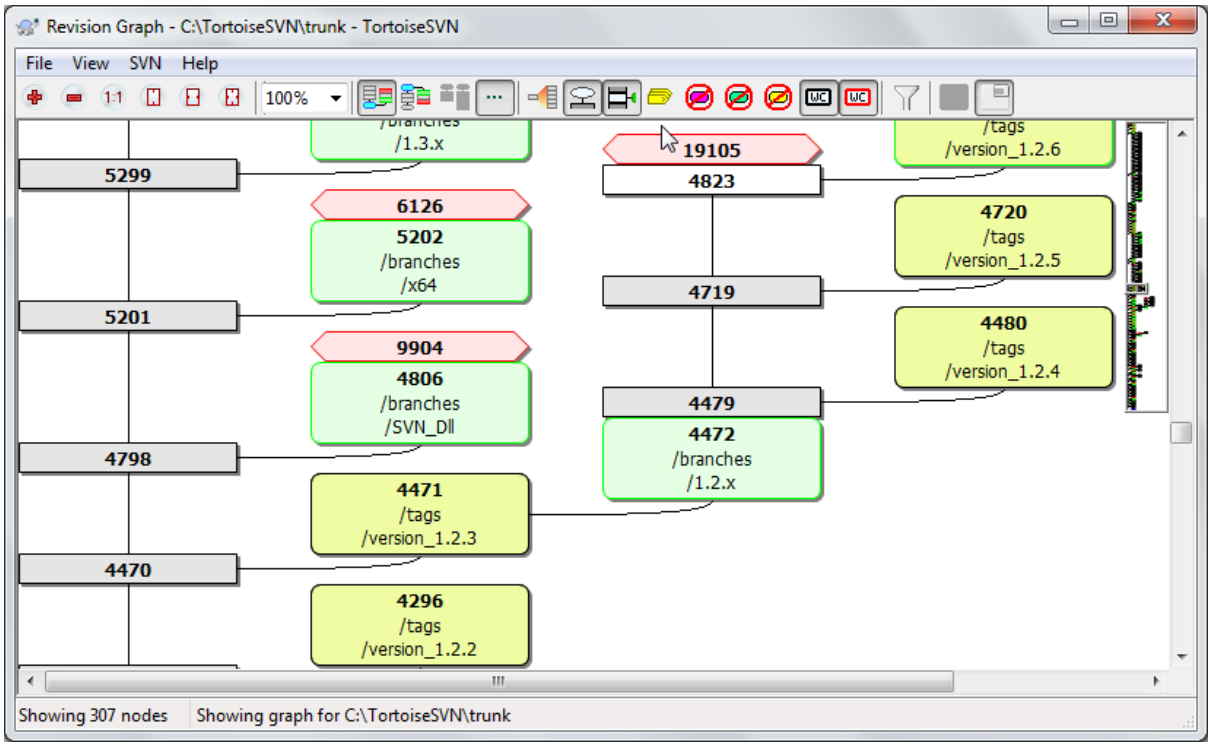
Če želite datoteko ali mapo premakniti/prekopirati na novo lokacijo in jo istočasno preimenovati, jo lahko povlečete z desnim gumbom (lahko držite pritisnjeno tipko **CTRL**) namesto z levim gumbom. V tem primeru se pokaže okno se vpis novega imena datoteke ali mape.

Ko ste naredili spremembe v skladišču z uporabo ene od teh metod, se pojavi okno za vnos sporočila dnevniškega zapisa. Če ste kaj premaknili po pomoti, imate sedaj možnost, da operacijo prekličete.

Včasih se pri odpiranju poti namesto vsebine prikaže sporočilo o napaki. Vzrok je lahko napačen naslov URL, nezadostne pravice ali težave s strežnikom. Če želite sporočilo o napaki prekopirati (da ga vključite v elektronsko sporočilo), iz kontekstnega menija izberite Kontekstni meni → Kopiraj sporočilo o napaki na odložišče ali uporabite **Ctrl+C**.

Bookmarked urls/repositories are shown below the current repository folders in the left tree view. You can add entries there by right clicking on any file or folder and select Context Menu → Add to Bookmarks. Clicking on a bookmark will browse to that repository and file/folder.

4.26. Grafi revizij



Slika 4.67. Graf revizije

Včasih želite izvedeti, od kje izhajajo veje in oznake in najlažji način, da to izveste, je graf revizije ali drevesne strukture. Takrat uporabite TortoiseSVN → Graf revizij...

Ta ukaz analizira zgodovino revizij in poskuša ustvariti drevo, ki prikazuje, kje so se zgodila kopiranja oziroma kje so bile veje/oznake izbrisane.

Pomembno

Za generiranje grafa mora TortoiseSVN iz strežnika prenesti dnevniške zapise vseh revizij. Verjetno ni potrebno poudarjati, da lahko pri skladiščih z nekaj tisoč revizijami to traja kar nekaj minut, odvisno tudi od hitrosti strežnika, pasovne širine omrežja... Če poskusite izdelati graf revizij na projektu, kot je denimo *Apache*, ki ima trenutno več kot pol milijona revizij, boste čakali kar lep čas.

The good news is that if you are using log caching, you only have to suffer this delay once. After that, log data is held locally. Log caching is enabled in TortoiseSVN's settings.

4.26.1. Vozlišča grafa revizij

Each revision graph node represents a revision in the repository where something changed in the tree you are looking at. Different types of node can be distinguished by shape and colour. The shapes are fixed, but colours can be set using TortoiseSVN → Settings

Added or copied items
: Items which have been added, or created by copying another file/folder are shown using a rounded rectangle. The default colour is green. Tags and trunks are treated as a special case and use a different shade, depending on the TortoiseSVN → Settings.

Deleted items
: Deleted items e.g. a branch which is no longer required, are shown using an octagon (rectangle with corners cut off). The default colour is red.

Renamed items
: Renamed items are also shown using an octagon, but the default colour is blue.

Revizija na vrhu veje
: The graph is normally restricted to showing branch points, but it is often useful to be able to see the respective HEAD revision for each branch too. If you select Show HEAD revisions, each HEAD revision nodes will be shown as an ellipse. Note that HEAD here refers to the last revision committed on that path, not to the HEAD revision of the repository.

Working copy revision
: If you invoked the revision graph from a working copy, you can opt to show the BASE revision on the graph using Show WC revision, which marks the BASE node with a bold outline.

Modified working copy
: If you invoked the revision graph from a working copy, you can opt to show an additional node representing your modified working copy using Show WC modifications. This is an elliptical node with a bold outline in red by default.

Normal item
: Vsi drugi elementi so prikazani s pravokotnikom

Note that by default the graph only shows the points at which items were added, copied or deleted. Showing every revision of a project will generate a very large graph for non-trivial cases. If you really want to see *all* revisions where changes were made, there is an option to do this in the View menu and on the toolbar.

The default view (grouping off) places the nodes such that their vertical position is in strict revision order, so you have a visual cue for the order in which things were done. Where two nodes are in the same column the order is very obvious. When two nodes are in adjacent columns the offset is much smaller because there is no need to prevent the nodes from overlapping, and as a result the order is a little less obvious. Such optimisations are necessary to keep complex graphs to a reasonable size. Please note that this ordering uses the *edge* of the node on the *older* side as a reference, i.e. the bottom edge of the node when the graph is shown with oldest node at the bottom. The reference edge is significant because the node shapes are not all the same height.

4.26.2. Spreminjanje pogleda

Ker je graf revizij pogosto kar zapleten, imate na voljo kar nekaj možnosti, da ga priredite po svojem okusu. Te možnosti so na voljo v meniju Pogled in v orodni vrstici.

Razdeljevanje vej v skupine
: The default behavior (grouping off) has all rows sorted strictly by revision. As a result, long-living branches with sparse commits occupy a whole column for only a few changes and the graph becomes very broad.

  This mode groups changes by branch, so that there is no global revision ordering: Consecutive revisions on a branch will be shown in (often) consecutive lines. Sub-branches, however, are arranged in such a way that later branches will be shown in the same column above earlier branches to keep the graph slim. As a result, a given row may contain changes from different revisions.

Starejše na vrhu
: Običajno graf prikazuje starejše revizije na dnu, drevo pa potem raste navzgor. Z vključitvijo te možnosti graf raste od vrha navzdol.

Poravnaj drevesa na vrhu
: When a graph is broken into several smaller trees, the trees may appear either in natural revision order, or aligned at the bottom of the window, depending on whether you are using the Group Branches option. Use this option to grow all trees down from the top instead.

Zmanjšaj križanje črt
: This option is normally enabled and avoids showing the graph with a lot of confused crossing lines. However this may also make the layout columns appear in less logical places, for example in a diagonal line rather than

a column, and the graph may require a larger area to draw. If this is a problem you can disable the option from the View menu.

Differential path names
: Long path names can take a lot of space and make the node boxes very large. Use this option to show only the changed part of a path, replacing the common part with dots. E.g. if you create a branch `/branches/1.2.x/doc/html` from `/trunk/doc/html` the branch could be shown in compact form as `/branches/1.2.x/..` because the last two levels, `doc` and `html`, did not change.

Show all revisions
: This does just what you expect and shows every revision where something (in the tree that you are graphing) has changed. For long histories this can produce a truly huge graph.

Pokaži revizijo HEAD
: S to možnostjo zagotovite, da je najnovejša revizija vsake veje vedno prikazana na grafu.

Natančni izvori kopiranja
: When a branch/tag is made, the default behaviour is to show the branch as taken from the last node where a change was made. Strictly speaking this is inaccurate since the branches are often made from the current HEAD rather than a specific revision. So it is possible to show the more correct (but less useful) revision that was used to create the copy. Note that this revision may be younger than the HEAD revision of the source branch.

Zloži oznake
: When a project has many tags, showing every tag as a separate node on the graph takes a lot of space and obscures the more interesting development branch structure. At the same time you may need to be able to access the tag content easily so that you can compare revisions. This option hides the nodes for tags and shows them instead in the tooltip for the node that they were copied from. A tag icon on the right side of the source node indicates that tags were made. This greatly simplifies the view.

  Note that if a tag is itself used as the source for a copy, perhaps a new branch based on a tag, then that tag will be shown as a separate node rather than folded.

Hide deleted paths
: Hides paths which are no longer present at the HEAD revision of the repository, e.g. deleted branches.

  If you have selected the Fold tags option then a deleted branch from which tags were taken will still be shown, otherwise the tags would disappear too. The last revision that was tagged will be shown in the colour used for deleted nodes instead of showing a separate deletion revision.

  If you select the Hide tags option then these branches will disappear again as they are not needed to show the tags.

Hide unused branches
: Hides branches where no changes were committed to the respective file or sub-folder. This does not necessarily indicate that the branch was not used, just that no changes were made to *this* part of it.

Show WC revision
: Marks the revision on the graph which corresponds to the update revision of the item you fetched the graph for. If you have just updated, this will be HEAD, but if others have committed changes since your last update your WC may be a few revisions lower down. The node is marked by giving it a bold outline.

Show WC modifications
: If your WC contains local changes, this option draws it as a separate elliptical node, linked back to the node that your WC was last updated to. The default outline colour is red. You may need to refresh the graph using **F5** to capture recent changes.

Filter
: Včasih graf revizij prikaže več podrobnosti, kot si želite. Ta možnost odpre pogovorno okno, ki omogoča omejevanje prikazanega območja revizij in skrivanje določenih poti po imenu.

If you hide a particular path and that node has child nodes, the children will be shown as a separate tree. If you want to hide all children as well, use the Remove the whole subtree(s) checkbox.

Tree stripes
: Where the graph contains several trees, it is sometimes useful to use alternating colours on the background to help distinguish between trees.

Show overview
: Shows a small picture of the entire graph, with the current view window as a rectangle which you can drag. This allows you to navigate the graph more easily. Note that for very large graphs the overview may become useless due to the extreme zoom factor and will therefore not be shown in such cases.

4.26.3. Uporaba grafa revizij

Za lažjo navigacijo po grafu revizij uporabite predogledno okno. Ta prikaže celoten graf v majhnem oknu, pri tem pa je trenutno prikazan del posebej označen. Označen del lahko premikate in s tem spremenite pogled grafa.

Datum revizije, avtor in komentarji so prikazani v namigu, ko miško pridržite na polju revizije.

Če izberete dve reviziji (uporabite **CTRL)**-levi klik), lahko uporabite kontekstni meni, da prikažete razlike med revizijama. Prikažete lahko tudi razlike med začetnimi točkami vej, vendar običajno prikazujemo razlike med končnimi točkami vej (revizije HEAD).

Razlike lahko prikažete kot datoteko poenotene različice, ki prikaže vse razlike v eni datoteki z minimalno vsebino. Če izberete Kontekstni meni → Primerjaj reviziji, se vam prikaže seznam spremenjenih datotek. Dvokliknite na datoteko, da prenesete obe reviziji datoteke in ju primerjate z grafičnim orodjem za razlikovanje.

Če kliknete z desnim gumbom na revizijo, lahko uporabite Kontekstni meni → Pokaži dnevnik.

You can also merge changes in the selected revision(s) into a different working copy. A folder selection dialog allows you to choose the working copy to merge into, but after that there is no confirmation dialog, nor any opportunity to try a test merge. It is a good idea to merge into an unmodified working copy so that you can revert the changes if it doesn't work out! This is a useful feature if you want to merge selected revisions from one branch to another.

Learn to Read the Revision Graph

First-time users may be surprised by the fact that the revision graph shows something that does not match the user's mental model. If a revision changes multiple copies or branches of a file or folder, for instance, then there will be multiple nodes for that single revision. It is a good practice to start with the leftmost options in the toolbar and customize the graph step-by-step until it comes close to your mental model.

All filter options try lose as little information as possible. That may cause some nodes to change their color, for instance. Whenever the result is unexpected, undo the last filter operation and try to understand what is special about that particular revision or branch. In most cases, the initially expected outcome of the filter operation would either be inaccurate or misleading.

4.26.4. Osveževanje pogleda

Če želite preveriti, ali obstajajo na strežniku novejše informacije, lahko pogled osvežite z uporabe tipke **F5**. Če uporabljate predpomnilnik dnevnika (privzeta nastavitev), s tem preverite, ali se na skladišču nahajajo nove objave, in prenesete le najnovejše zapise. Če ste pred delali v nepovezanem načinu, bo TortoiseSVN poskušal vzpostaviti povezavo.

Če uporabljate predpomnilnik dnevnika in menite, da se je sporočilo dnevniškega zapisa ali avtor spremenil, uporabite okno dnevnika za osvežitev željenih sporočil. Ker se graf revizij sestavlja iz korena skladišča, bi bilo potrebno preveriti celoten predpomnilnik. Posodabljanje bi trajalo *zelo* dolgo.

4.26.5. Pruning Trees

A large tree can be difficult to navigate and sometimes you will want to hide parts of it, or break it down into a forest of smaller trees. If you hover the mouse over the point where a node link enters or leaves the node you will see one or more popup buttons which allow you to do this.

Click on the minus button to collapse the attached sub-tree.

Click on the plus button to expand a collapsed tree. When a tree has been collapsed, this button remains visible to indicate the hidden sub-tree.

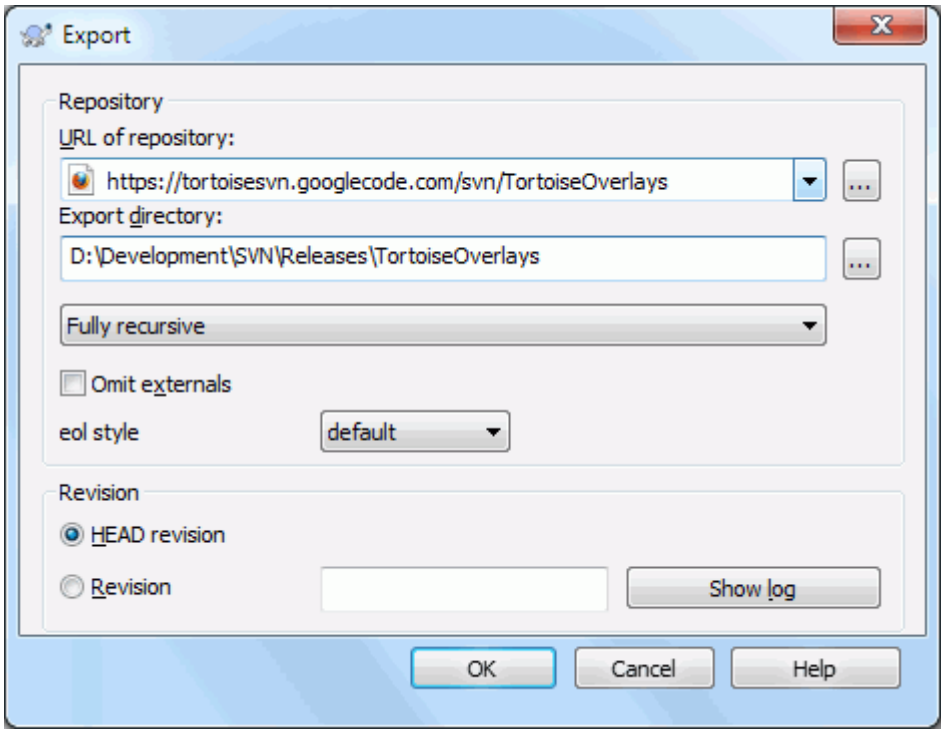
Click on the cross button to split the attached sub-tree and show it as a separate tree on the graph.

Click on the circle button to reattach a split tree. When a tree has been split away, this button remains visible to indicate that there is a separate sub-tree.

Click on the graph background for the main context menu, which offers options to Expand all and Join all. If no branch has been collapsed or split, the context menu will not be shown.

4.27. Izvažanje delovne kopije sistema Subversion

Sometimes you may want a clean copy of your working tree without the `.svn` directory, e.g. to create a zipped tarball of your source, or to export to a web server. Instead of making a copy and then deleting the `.svn` directory manually, TortoiseSVN offers the command TortoiseSVN → Export.... Exporting from a URL and exporting from a working copy are treated slightly differently.



Slika 4.68. Okno Uvoz-iz-URL

Če izvedete ta ukaz na mapi brez različic TortoiseSVN predpostavlja, da je izbrana mapa cilj, in odpre pogovorno okno za vnos naslova URL in številke revizije za izvoz. Okno omogoča izvoz samo vrhnje mape, omogoča izpustitev zunanjih referenc, prav tako pa omogoča spreminjanje sloga zaključevanja vrstic pri datotekah, ki imajo nastavljeno lastnost `svn:eol-style`.

Seveda lahko izvažate tudi neposredno iz skladišča. Z brskalnikom po skladišču pojdite na ustrezno poddrevo skladišča in uporabite Kontekstni meni → Izvozi. Prikazalo se bo okno Izvoz iz URL, kot je to opisano zgoraj.

If you execute this command on your working copy you'll be asked for a place to save the *clean* working copy without the `.svn` folder. By default, only the versioned files are exported, but you can use the Export unversioned files too checkbox to include any other unversioned files which exist in your WC and not in the repository. External references using `svn:externals` can be omitted if required.

Another way to export from a working copy is to right drag the working copy folder to another location and choose Context Menu → SVN Export versioned items here or Context Menu → SVN Export all items here or Context Menu → SVN Export changed items here. The second option includes the unversioned files as well. The third option exports only modified items, but maintains the folder structure.

When exporting from a working copy, if the target folder already contains a folder of the same name as the one you are exporting, you will be given the option to overwrite the existing content, or to create a new folder with an automatically generated name, e.g. `Target (1)`.

Izvažanje posameznih datotek

Pogovorno okno za izvažanje ne dovoli izvažanja posameznih datotek, čeprav je Subversion to zmožen narediti.

Za izvoz posamezne datoteko s programom TortoiseSVN, morate uporabiti brskalnik po skladišču (Razdelek 4.25, “Brskalnik po skladišču”). Datoteko, ki jo želite izvoziti, preprosto odvlečite v Raziskovalca ali pa uporabite kontekstni meni.

Exporting a Change Tree

If you want to export a copy of your project tree structure but containing only the files which have changed in a particular revision, or between any two revisions, use the compare revisions feature described in Razdelek 4.11.3, “Primerjanje map”.

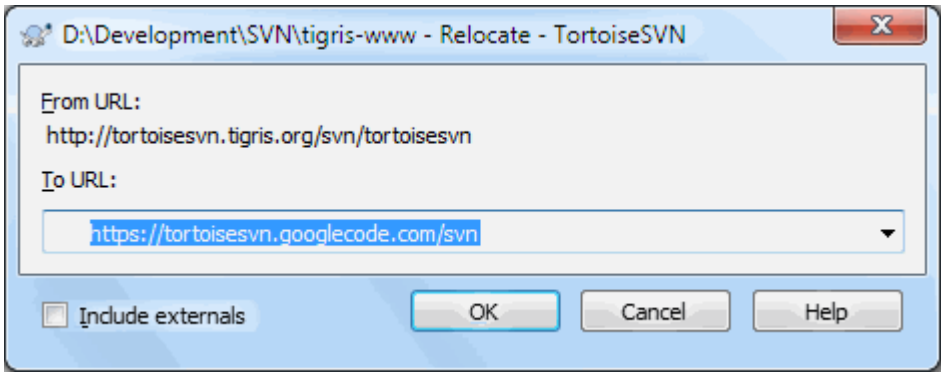
If you want to export your working copy tree structure but containing only the files which are locally modified, refer to SVN Export changed items here above.

4.27.1. Kako odstranim delovno kopijo iz nadzora različic

Sometimes you have a working copy which you want to convert back to a normal folder without the `.svn` directory. All you need to do is delete the `.svn` directory from the working copy root.

Alternatively you can export the folder to itself. In Windows Explorer right drag the working copy root folder from the file pane onto itself in the folder pane. TortoiseSVN detects this special case and asks if you want to make the working copy unversioned. If you answer *yes* the control directory will be removed and you will have a plain, unversioned directory tree.

4.28. Premeščanje delovne kopije



Slika 4.69. Okno za premeščanje

If your repository has for some reason changed it's location (IP/URL). Maybe you're even stuck and can't commit and you don't want to checkout your working copy again from the new location and to move all your changed data back into the new working copy, TortoiseSVN → Relocate is the command you are looking for. It basically does very little: it rewrites all URLs that are associated with each file and folder with the new URL.

Opomba

This operation only works on working copy *roots*. So the context menu entry is only shown for working copy roots.

You may be surprised to find that TortoiseSVN contacts the repository as part of this operation. All it is doing is performing some simple checks to make sure that the new URL really does refer to the same repository as the existing working copy.

Pozor

Ta operacija se uporablja zelo redko. Ukaz Premesti se uporablja *le*, če se spremeni naslov URL korenske mape skladišča. Možni razlogi za to so:

- Spremenil se je IP naslov strežnika.
- Spremenil se je protokol (n.pr. http:// v https://).
- Spremenila se je korenska pot skladišča.

Povedano drugače, ukaz Premesti uporabite takrat, ko vaša delovna kopija kaže na isto lokacijo v istem skladišču, se je pa samo skladišče preselilo.

Ne velja, če:

- se želite premakniti v drugo skladišče Subversion. V tem primeru je najbolje narediti svež prevzem iz novega skladišča.
- želite narediti preklop na drugo vejo ali mapo znotraj istega skladišča. Za to uporabite TortoiseSVN → Preklopi.... Za več informacij preberite Razdelek 4.20.3, “Prevzeti ali preklopiti...”.

Če uporabite ukaz Premesti v kateremkoli od zgoraj naštetih dveh primerih, *boste pokvarili svojo delovno kopijo* in prikazalo se vam bo veliko nerazložljivih napak pri objavljanju, posodabljanju... Če se vam to zgodi, je edina rešitev svež prevzem.

4.29. Integracija s sistemi za sledenje zadev

V razvoju programske opreme se pogosto dogaja, da so spremembe vezane na določenega hrošča ali številko zadeve. Uporabniki sistemov za sledenje zadev pogosto želijo asociirati spremembe v sistemu Subversion z določeno številko zadeve v sistemu za sledenje zadev. Večina sistemov za sledenje zadev omogoča pisanje ukaznih datotek akcije pred objavo, ki pregleda sporočilo dnevniškega zapisa in poišče številko zadeve. Tak način je podvržen napakam, saj se zanaša na dejstvo, da bodo uporabniki napisali sporočilo dnevniškega zapisa v pravilni obliki.

TortoiseSVN lahko pomaga uporabniku na dva načina:

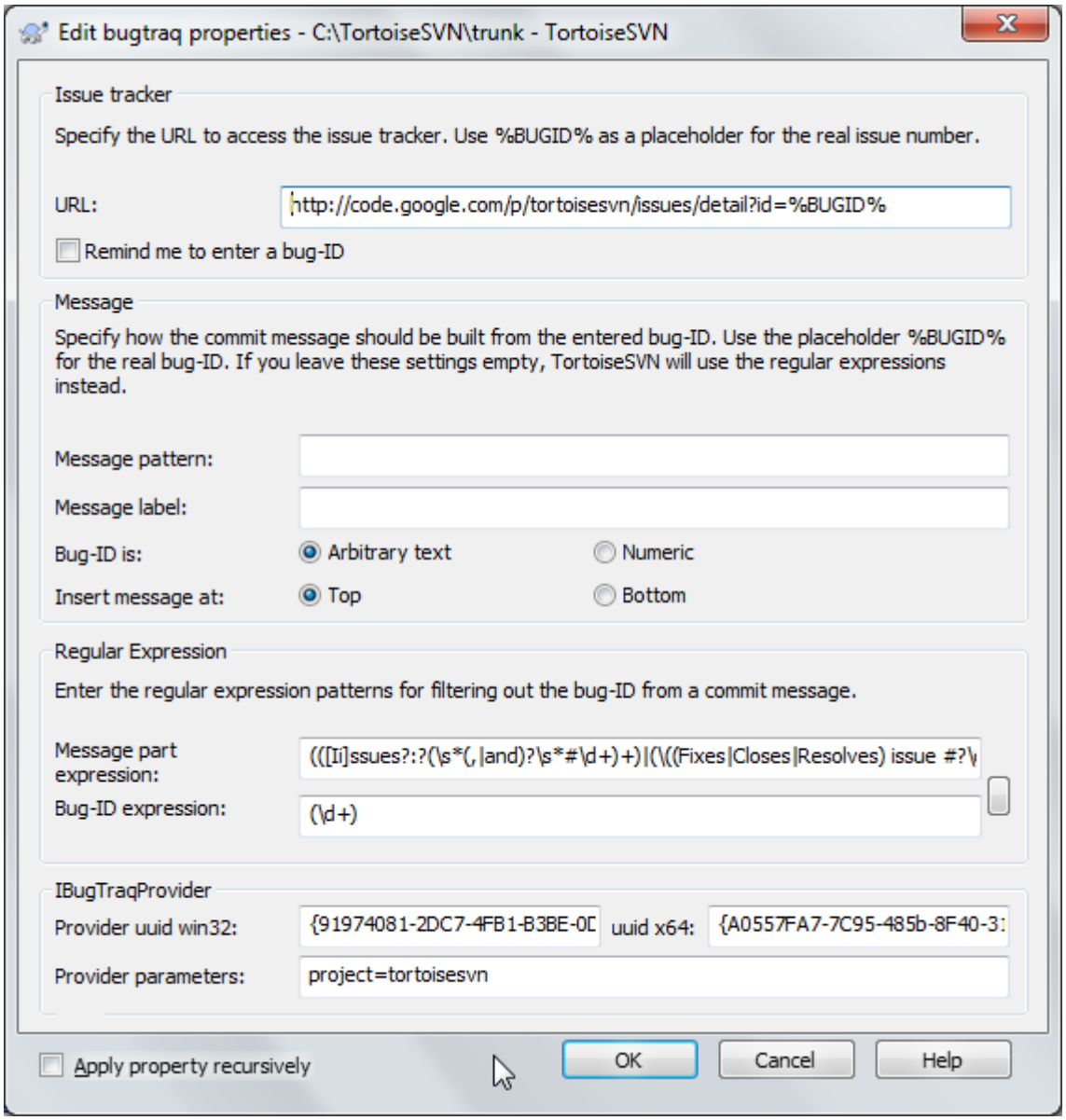
1. Ko uporabnik vpiše sporočilo dnevniškega zapisa, lahko samodejno doda predhodno določeno vrstico, ki vsebuje številko zadeve, ki se nanaša na objavo. Tako se zmanjša možnost, da uporabnik vpiše številko zadeve v obliki, ki jo orodje za sledenje zadev ne zna pravilno razčleniti.

   Lahko pa TortoiseSVN označi del vnešenega sporočila dnevniškega zapisa, ki ga sledilnik zadev prepozna. Tako uporabnik ve, da je sporočilo dnevniškega zapisa mogoče pravilno razčleniti.

2. Ko uporabnik brska po dnevniških zapisih, TortoiseSVN ustvari povezavo za vsako številko hrošča v sporočilu dnevniškega zapisa, ki požene spletni brskalnik in naloži stran ustrezne zadeve.

4.29.1. Dodajanje številk zadev dnevniškim zapisom

V TortoiseSVN lahko integrirate sledilnik zadev po vaši izbiri. Za to morate definirati nekaj lastnosti, ki se začnejo z `bugtraq:`. Nastavljene morajo biti na mapah: (Razdelek 4.18, “Nastavitve projekta”)



Slika 4.70. The Bugtraq Properties Dialog

When you edit any of the bugtraq properties a special property editor is used to make it easier to set appropriate values.

V TortoiseSVN lahko integrirate sisteme za sledenje na dva načina. Prvi je na osnovi enostavnih nizov, drugi pa na osnovi *regularnih izrazov*. Lastnosti, ki se uporabljajo pri obeh pristopih, so:

bugtraq:url
: Set this property to the URL of your bug tracking tool. It must be properly URI encoded and it has to contain `%BUGID%`. `%BUGID%` is replaced with the Issue number you entered. This allows TortoiseSVN to display a link in the log dialog, so when you are looking at the revision log you can jump directly to your bug tracking tool. You do not have to provide this property, but then TortoiseSVN shows only the issue number and not the link to it. e.g the TortoiseSVN project is using `http://issues.tortoisesvn.net/?do=details&id=%BUGID%`.

  Namesto absolutnih lahko uporabljate tudi relativne naslove URL. To je uporabno v primeru, ko je sledilnik zadev na isti domeni/strežniku kot skladišče Subversion. Če se ime domeni kdaj spremeni, vam tako ni potrebno spreminjati lastnosti `bugtraq:url`. Obstajata dva načina, kako določiti relativen naslov URL:

Če se začne z nizom `^/`, to pomeni pot, relativno glede na korensko mapo skadišča. Primer: `^/../?do=details&id=%BUGID%` se razširi v `http://tortoisesvn.net/?do=details&id=%BUGID%` če se skladišče nahaja na `http://tortoisesvn.net/svn/trunk/`.

Naslov URL, ki se začne z nizom `/`, je relativen na ime strežnika. Primer: `/?do=details&id=%BUGID%` se razširi v `http://tortoisesvn.net/?do=details&id=%BUGID%`, če se skladišče nahaja kjerkoli na naslovu `http://tortoisesvn.net`.

bugtraq:warnifnoissue
Nastavite to vrednost na `true`, če želite, da vas TortoiseSVN opozori, če pustite polje za vnos številke zadeve prazno. Dovoljene vrednosti so `true/false`. *Če vrednost ni nastavljena, se predpostavlja `false` (brez opozarjanja).*

4.29.1.1. Številka zadeve v vnosnem polju

Pri enostavnem načinu TortoiseSVN uporabniku pokaže posebno polje za vnos številke hrošča. Nato se na začetek/konec dnevniškega zapisa doda posebna vrstica.

bugtraq:message
Ta lastnost aktivira način *z vnosnim poljem*. Če nastavite to lastnost, vas TortoiseSVN ob objavi povpraša po številki zadeve. Številka se vnese na konec sporočila dnevniškega zapisa. Vsebovati mora niz `%BUGID%`, ki se ob objavi zamenja s številko zadeve. To zagotavlja, da sporočilo vedno vsebuje referenco na številko zadeve, ki je podana v pravilni obliki in jo orodje za sledenje zadev lahko razčleni ter poveže s to objavo. Primer: projekt TortoiseSVN uporablja vrednost `Issue : %BUGID%`, vendar je to odvisno od orodja, ki ga uporabljate.

bugtraq:label
To besedilo se prikaže v oknu za objave in opisuje vnosno polje za vpis številke zadeve. Če ni nastavljeno, se pojavi napis `Bug-ID / Issue-Nr:`. Upoštevajte, da se okno ne bo samodejno povečalo, zato omejite število znakov v napisu na 20 do 25.

bugtraq:number
Če je vrednost nastavljena na `true`, je v polje za številko zadeve dovoljeno vpisati le številke. Izjema je le vejica, ki jo uporabite, da vnesete več številk zadeve. Veljavni vrednosti sta `true/false`. *Če vrednost ni nastavljena, se predpostavlja `true`.*

bugtraq:append
Ta lastnost določa, ali se številka hrošča (zadeve) pripne na konec sporočila dnevniškega zapisa (true) ali na začetek (false). Veljavni vrednosti sta `true/false`. *Če vrednost ni nastavljena, se predpostavlja `true`, zato da obstojoči projekti še vedno delujejo.*

4.29.1.2. Številka zadeve z uporabo regularnih izrazov

Pri pristopu z *regularnimi izrazi* TortoiseSVN ne prikaže vnosnega polja, ampak označi del sporočila dnevniškega zapisa, ki ga orodje za sledenje zadev prepozna. To se dogaja med vnosom besedila. To pomeni tudi, da se številka zadeve lahko nahaja kjerkoli v sporočilu. Ta metoda je precej bolj fleksibilna in jo uporablja tudi sam projekt TortoiseSVN.

bugtraq:logregex
Ta lastnost aktivira sistem sledenja zadev v načinu *Regex*. Vsebuje enega ali dva regularna izraza, ločena z novo vrstico.

If two expressions are set, then the first expression is used as a pre-filter to find expressions which contain bug IDs. The second expression then extracts the bare bug IDs from the result of the first regex. This allows you to use a list of bug IDs and natural language expressions if you wish. e.g. you might fix several bugs and include a string something like this: “This change resolves issues #23, #24 and #25”.

If you want to catch bug IDs as used in the expression above inside a log message, you could use the following regex strings, which are the ones used by the TortoiseSVN project: `[Ii]ssues?:?(\s*(,|and)?\s*#\d+)+` and `(\d+)`.

Prvi izraz iz sporočila dnevniškega zapisa izlušči niz “napake #23, #24 in #25”. Drugi regularni izraz iz rezultata prvega izraza izlušči številke zadev. V tem primeru vrne “23”, “24” in “25”, ki se uporabijo kot številke zadev.

Če malce razčlenimo prvi regularni izraz: začeti se mora z besedo “napaka”. Beseda se lahko začne z veliko začetnico, lahko pa je zapisana v obliki dvojine ali množine (“[aie]”). Besedi lahko sledi dvopičje. Sledi ena ali več skupin, vsaka lahko vsebuje vodilne presledke, vejico ali niz “in” in zaključne presledke. Na koncu se nahaja obvezen znak “#” in obvezna številka.

Če je nastavljen le en izraz, potem mora le številka zadeve ustrezati skupini v regularnemu izrazu. Primer: `[Nn]apaka(?:)? #?(\d+)` To metodo zahtevajo nekateri sledilniki zadev, n. pr. trac, in jih je težje zgraditi. Zato priporočamo, da to metodo uporabite le, če tako velevajo navodila izbranega sledilnika zadev.

If you are unfamiliar with regular expressions, take a look at the introduction at https://en.wikipedia.org/wiki/Regular_expression, and the online documentation and tutorial at http://www.regular-expressions.info/.

It's not always easy to get the regex right so to help out there is a test dialog built into the bugtraq properties dialog. Click on the button to the right of the edit boxes to bring it up. Here you can enter some test text, and change each regex to see the results. If the regex is invalid the edit box background changes to red.

Če sta nastavljeni obe lastnosti - `bugtraq:message` in `bugtraq:logregex` - se upošteva `logregex`.

Namig

Tudi če nimate orodja za sledenje zadev, ki ga nastavite v skripto akcij za akcijo pred objavo, lahko še vedno uporabljate ta sistem, da številke zadev v sporočilu spremenite v povezave!

Povezav mogoče niti ne potrebujete, številke zadev pa se vseeno pojavijo v posebnem stolpcu v dnevniku, tako da lahko hitreje najdete spremembe, ki so povezane z določeno zadevo.

Nekatere lastnosti `tsvn:` zahtevajo vrednost `true/false`. TortoiseSVN razume tudi besedo `yes` kot sinonim za `true` in besedo `no` kot sinonim za `false`.

Nastavite lastnosti na mapah

These properties must be set on folders for the system to work. When you commit a file or folder the properties are read from that folder. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (e.g. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it's enough if you set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. A property setting deeper in the project hierarchy overrides settings on higher levels (closer to `trunk/`).

As of version 1.8, TortoiseSVN and Subversion use so called `inherited properties`, which means a property that is set on a folder is automatically also implicitly set on all subfolders. So there's no need to set the properties on all folders anymore but only on the root folder.

For project properties *only*, i.e. `tsvn:`, `bugtraq:` and `webviewer:` you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

When you add new sub-folders to a working copy using TortoiseSVN, any project properties present in the parent folder will automatically be added to the new child folder too.

No Issue Tracker Information from Repository Browser

Because the issue tracker integration depends upon accessing Subversion properties, you will only see the results when using a checked out working copy. Fetching properties remotely is a slow

> operation, so you will not see this feature in action from the repo browser unless you started the repo browser from your working copy. If you started the repo browser by entering the URL of the repository you won't see this feature.
>
> For the same reason, project properties will not be propagated automatically when a child folder is added using the repo browser.

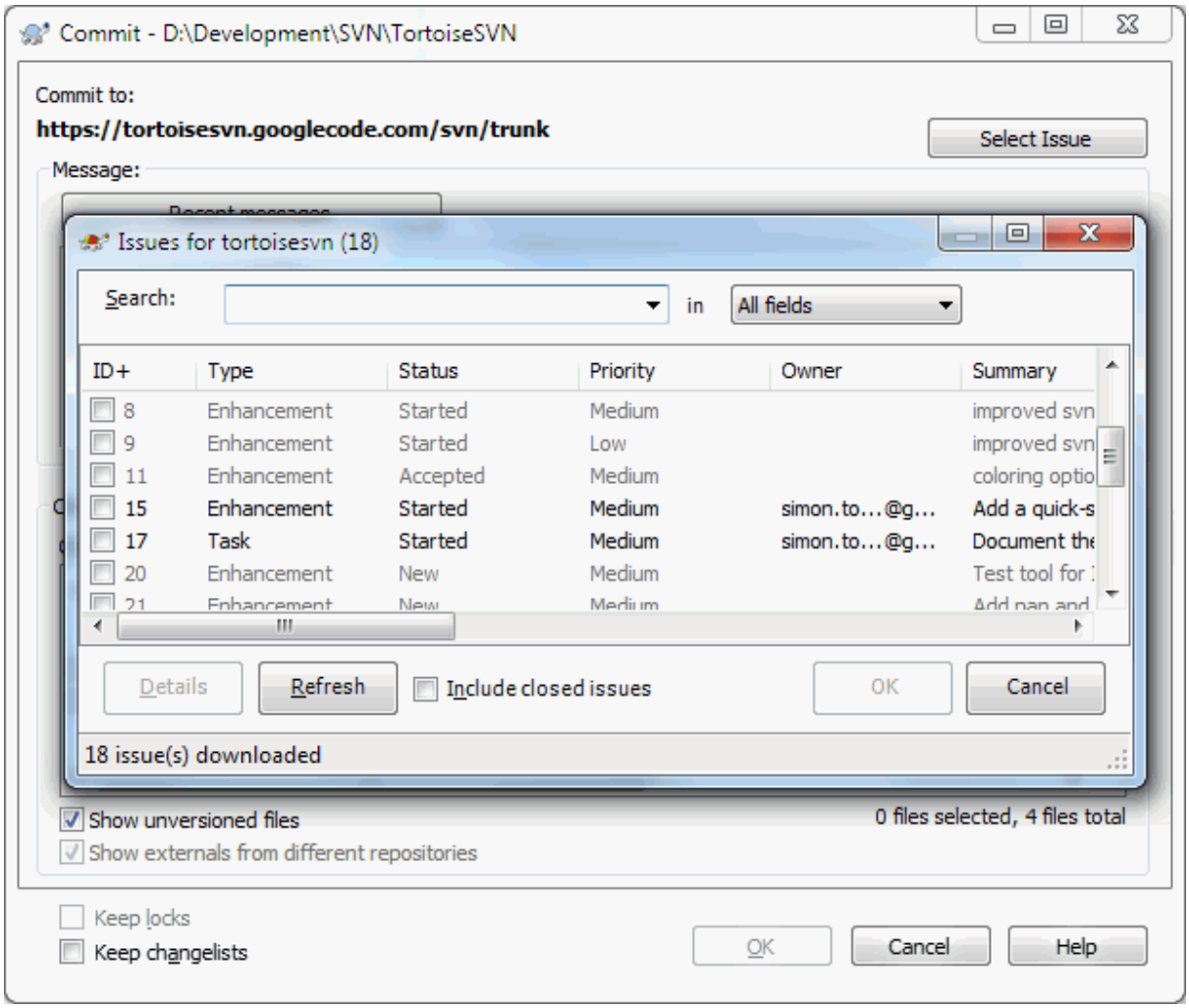
This issue tracker integration is not restricted to TortoiseSVN; it can be used with any Subversion client. For more information, read the full *Issue Tracker Integration Specification* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/doc/notes/issuetrackers.txt] in the TortoiseSVN source repository. (Razdelek 3, “License” explains how to access the repository.)

4.29.2. Pridobivanje informacij iz sledilnika zadev

The previous section deals with adding issue information to the log messages. But what if you need to get information from the issue tracker? The commit dialog has a COM interface which allows integration an external program that can talk to your tracker. Typically you might want to query the tracker to get a list of open issues assigned to you, so that you can pick the issues that are being addressed in this commit.

Any such interface is of course highly specific to your issue tracker system, so we cannot provide this part, and describing how to create such a program is beyond the scope of this manual. The interface definition and sample plugins in C# and C++/ATL can be obtained from the `contrib` folder in the *TortoiseSVN repository* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/issue-tracker-plugins]. (Razdelek 3, “License” explains how to access the repository.) A summary of the API is also given in Poglavje 7, *IBugtraqProvider interface*. Another (working) example plugin in C# is *Gurtle* [http://code.google.com/p/gurtle/] which implements the required COM interface to interact with the *Google Code* [http://code.google.com/hosting/] issue tracker.

Za ilustracijo si predstavljajmo, da vam je skrbnik sistema priskrbel vtičnik za sledilnik zadev, ki ste ga namestili, sedaj pa ga morate uporabiti pri delovnih kopijah. Vse potrebno postorite v oknu Nastavitve programa TortoiseSVN. Ko odprete okno za objave v delovni kopiji, ki ima nastavljen vtičnik, se na vrhu pogovornega okna pojavi nov gumb.



Slika 4.71. Primer poizvedovalnika sledilnika zadev

V tem primeru lahko izberete eno ali več odprtih zadev. Vtičnik nato ustvari posebej oblikovano besedilo, ki ga doda v sporočilo dnevniškega zapisa.

4.30. Integracija z internetno naravnanimi pregledovalniki skladišč

Obstaja kar nekaj spletnih pregledovalnikov skladišč za Subversion, n. pr. *ViewVC* [http://www.viewvc.org/] in *WebSVN* [http://websvn.tigris.org/]. TortoiseSVN nudi možnost povezave s temi pregledovalniki.

V TortoiseSVN lahko integrirate pregledovalnik skladišč po izbiri. Za to morate nastaviti nekaj lastnosti, ki definirajo povezavo. Nastavljene morajo biti na mapah: (Razdelek 4.18, “Nastavitve projekta”)

webviewer:revision
: Set this property to the URL of your repo viewer to view all changes in a specific revision. It must be properly URI encoded and it has to contain `%REVISION%`. `%REVISION%` is replaced with the revision number in question. This allows TortoiseSVN to display a context menu entry in the log dialog Context Menu → View revision in webviewer.

webviewer:pathrevision
: Set this property to the URL of your repo viewer to view changes to a specific file in a specific revision. It must be properly URI encoded and it has to contain `%REVISION%` and `%PATH%`. `%PATH%` is replaced with the path relative to the repository root. This allows TortoiseSVN to display a context menu entry in the log dialog Context Menu → View revision for path in webviewer For example, if you right click in the log

dialog bottom pane on a file entry `/trunk/src/file` then the `%PATH%` in the URL will be replaced with `/trunk/src/file`.

Namesto absolutnih naslovov URL lahko uporabljate tudi relativne. To je uporabno, če se vaš spletni brskalnik po skladišču nahaja na isti domeni/strežniku kot skladišče. Če se ime domene kadarkoli spremeni, vam lastnosti `webviewer:revision` in `webviewer:pathrevision` ni potrebno spreminjati. Oblika je enaka kot za lastnost `bugtraq:url`. Poglejte Razdelek 4.29, “Integracija s sistemi za sledenje zadev”.

Nastavite lastnosti na mapah

These properties must be set on folders for the system to work. When you commit a file or folder the properties are read from that folder. If the properties are not found there, TortoiseSVN will search upwards through the folder tree to find them until it comes to an unversioned folder, or the tree root (e.g. `C:\`) is found. If you can be sure that each user checks out only from e.g `trunk/` and not some sub-folder, then it's enough if you set the properties on `trunk/`. If you can't be sure, you should set the properties recursively on each sub-folder. A property setting deeper in the project hierarchy overrides settings on higher levels (closer to `trunk/`).

For project properties *only*, i.e. `tsvn:`, `bugtraq:` and `webviewer:` you can use the Recursive checkbox to set the property to all sub-folders in the hierarchy, without also setting it on all files.

When you add new sub-folders to a working copy using TortoiseSVN, any project properties present in the parent folder will automatically be added to the new child folder too.

Limitations Using the Repository Browser

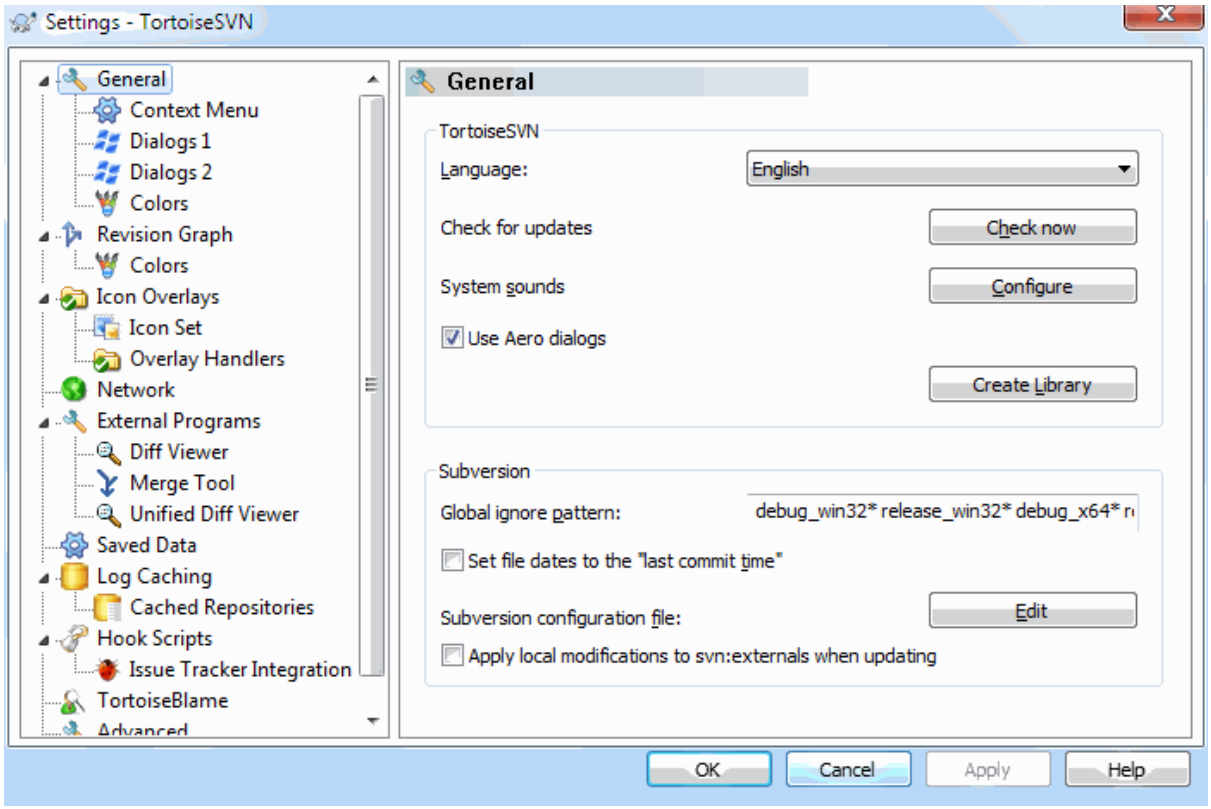
Because the repo viewer integration depends upon accessing Subversion properties, you will only see the results when using a checked out working copy. Fetching properties remotely is a slow operation, so you will not see this feature in action from the repo browser unless you started the repo browser from your working copy. If you started the repo browser by entering the URL of the repository you won't see this feature.

For the same reason, project properties will not be propagated automatically when a child folder is added using the repo browser.

4.31. Nastavitve TortoiseSVN

Da izveste, čemu služi določena nastavitev, počakajte z miško nekaj trenutkov nad vnosnim/potrditvenim poljem... in prikazal se vam bo uporaben namig.

4.31.1. Splošne nastavitve



Slika 4.72. Okno za nastavitve, Splošno

To okno vam omogoča nastavitev jezika in nastavitev, specifičnih za Subversion.

Jezik
: Selects your user interface language. Of course, you have to install the corresponding language pack first to get another UI language than the default English one.

Preveri posodobitve
: TortoiseSVN will contact its download site periodically to see if there is a newer version of the program available. If there is it will show a notification link in the commit dialog. Use Check now if you want an answer right away. The new version will not be downloaded; you simply receive an information dialog telling you that the new version is available.

Sistemski zvoki
: TortoiseSVN ima tri lastne zvočne efekte, ki so vedno nameščeni.

  - Napaka
  - Obvestilo
  - Opozorilo

  Preko Nadzorne plošče sistema Windows lahko izberete zvoke (ali popolnoma izklopite privzete). Gumb Nastavi je bližnjica do Nadzorne plošče.

Use Aero Dialogs
: On Windows Vista and later systems this controls whether dialogs use the Aero styling.

Create Library
: On Windows 7 you can create a Library in which to group working copies which are scattered in various places on your system.

Splošni vzorec prezrtih

Global ignore patterns are used to prevent unversioned files from showing up e.g. in the commit dialog. Files matching the patterns are also ignored by an import. Ignore files or directories by typing in the names or extensions. Patterns are separated by spaces e.g. `bin obj *.bak *.~?? *.jar *.[Tt]mp`. These patterns should not include any path separators. Note also that there is no way to differentiate between files and directories. Read Razdelek 4.14.1, “Iskanje vzorcev v seznamu prezrtih elementov” for more information on the pattern-matching syntax.

Upoštevajte, da bodo vzorci prezrtih elementov, ki jih nastavite tukaj, vplivali tudi na druge odjemalce za Subversion, ki so nameščeni na vašem računalniku, vključno z odjemalcem za ukazno vrstico.

Opozorilo

Če uporabljate konfiguracijsko datoteko Subversion za nastavljanje `splošnega vzorca prezrtih`, bodo nastavitve, ki so nastavite tu, povožene. Do konfiguracijske datoteke Subversion dostopate z gumbom Uredi, kot je opisano spodaj.

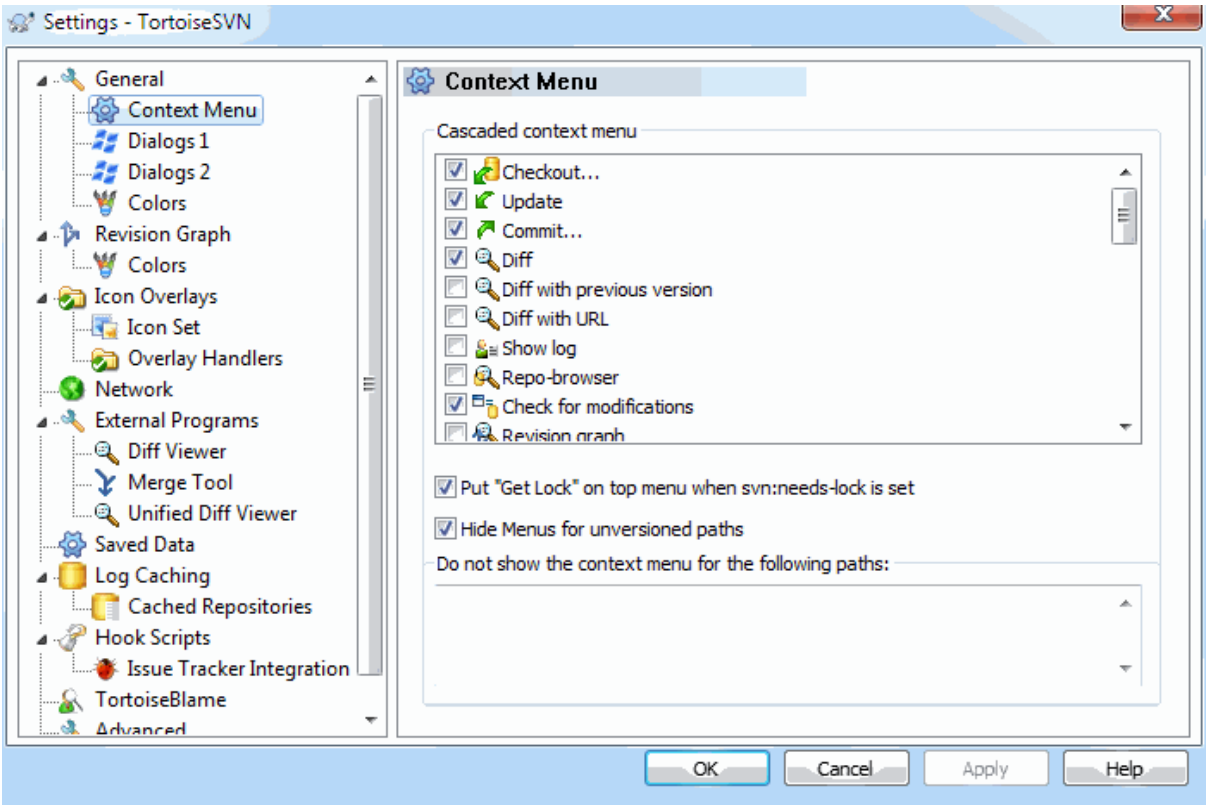
This ignore pattern will affect all your projects. It is not versioned, so it will not affect other users. By contrast you can also use the versioned `svn:ignore` or `svn:global-ignores` property to exclude files or directories from version control. Read Razdelek 4.14, “Dodajanje datotek in map na seznam prezrtih elementov” for more information.

Nastavi datume datotek na “čas zadnje objave”
: Ta možnost pove sistemu TortoiseSVN, naj pri prevzemu ali posodobitvi popravi časovni žig datotek na čas objave. V nasprotnem primeru TortoiseSVN uporabi trenutni datum. Če razvijate programsko opremo, je najbolje, da uporabljate trenuten datum/čas, saj sistemi za gradjo običajno pogledajo datum in čas spremembe datoteke, da ugotovijo, katere je potrebno ponovno prevesti. Če uporabite “zadnji čas objave” in naredite povrnitev, se projekt morda ne bo prevedel tako, kot želite.

Nastavitvena datoteka Subversion
: Use Edit to edit the Subversion configuration file directly. Some settings cannot be modified directly by TortoiseSVN, and need to be set here instead. For more information about the Subversion `config` file see the *Runtime Configuration Area* [http://svnbook.red-bean.com/en/1.8/svn.advanced.confarea.html]. The section on *Automatic Property Setting* [http://svnbook.red-bean.com/en/1.8/svn.advanced.props.html#svn.advanced.props.auto] is of particular interest, and that is configured here. Note that Subversion can read configuration information from several places, and you need to know which one takes priority. Refer to *Configuration and the Windows Registry* [http://svnbook.red-bean.com/en/1.8/svn.advanced.confarea.html#svn.advanced.confarea.windows-registry] to find out more.

Apply local modifications to svn:externals when updating
: This option tells TortoiseSVN to always apply local modifications to the `svn:externals` property when updating the working copy.

4.31.1.1. Nastavitve kontekstnega menija



Slika 4.73. Okno nastavitev, Kontekstni meni

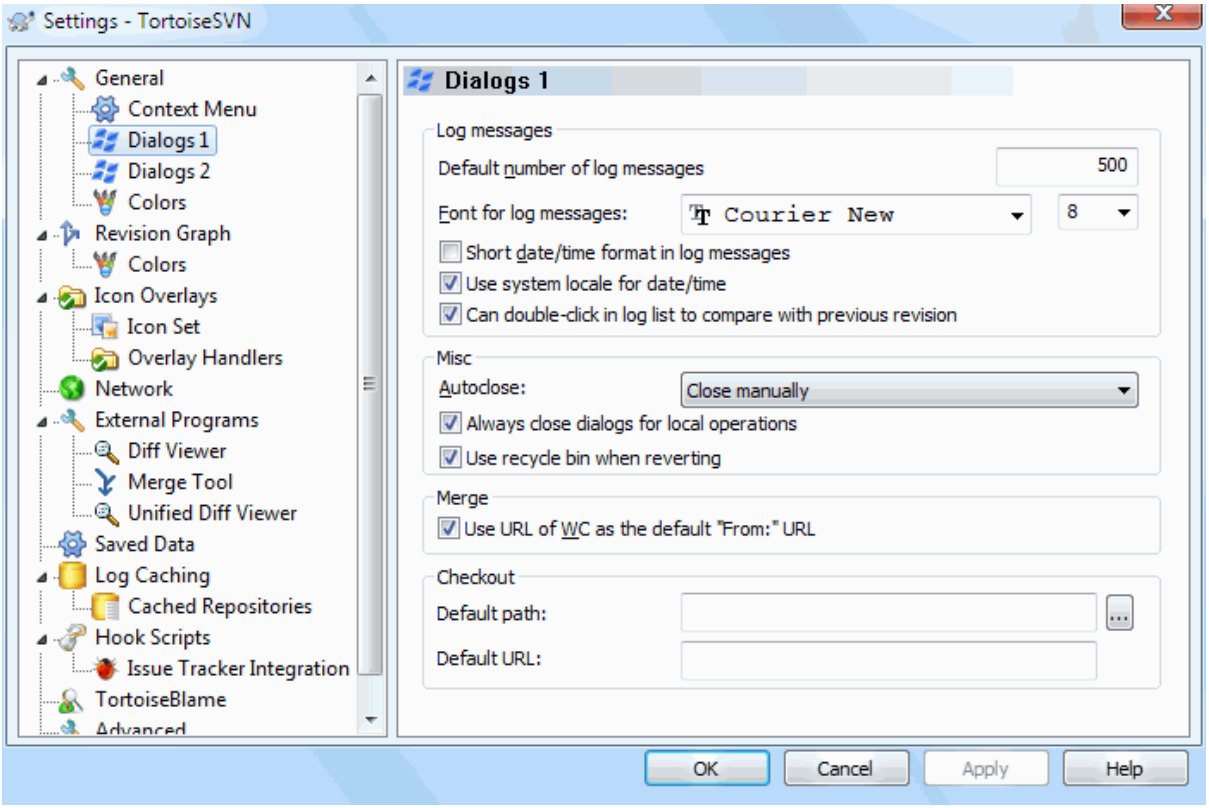
Ta stran omogoča nastavitev vnosov kontekstnega menija TortoiseSVN, ki se pojavijo v glavnem kontekstnem meniju. Po privzetih nastavitvah večina vnosov ni izbranih in se pojavijo samo v podmeniju.

Ukaz Dobi zaklep je poseben primer. Seveda ga lahko postavite v glavni kontekstni meni, kot je opisano zgoraj. Ker pa večina datotek ne potrebuje zaklepanja, je to le nepotrebna obremenitev menija. Po drugi strani pa pri datotekah, ki imajo nastavljeno lastnost `svn:needs-lock`, potrebujete ta ukaz pri vsakem urejanju, zato je v tem primeru koristno, če se nahaja v glavnem kontekstnem meniju. Če potrdite potrditveno polje, se bo ukaz Dobi zaklep pojavil v glavnem meniju vsakokrat, ko izberete datoteko z nastavljeno lastnostjo `svn:needs-lock`.

Most of the time, you won't need the TortoiseSVN context menu, apart for folders that are under version control by Subversion. For non- versioned folders, you only really need the context menu when you want to do a checkout. If you check the option `Hide menus for unversioned paths` , TortoiseSVN will not add its entries to the context menu for unversioned folders. But the entries are added for all items and paths in a versioned folder. And you can get the entries back for unversioned folders by holding the **Shift** key down while showing the context menu.

If there are some paths on your computer where you just don't want TortoiseSVN's context menu to appear at all, you can list them in the box at the bottom.

4.31.1.2. Pogovorna okna 1



Slika 4.74. Okno nastavitev, Pogovorna okna 1

To okno omogoča nastavitev nekaterih pogovornih oken programa TortoiseSVN po vaših željah.

Privzeto število dnevniških zapisov
: Omeji število dnevniških zapisov, ki jih TortoiseSVN prenese, ko prvič izberete TortoiseSVN → Pokaži dnevnik. Uporabno za počasne povezave. Vedno lahko uporabite gumb Pokaži vse ali gumb Naslednjih 100, če želite videti več zapisov.

Pisava za dnevniški zapis
: Izbere pisavo in velikost, ki se uporablja za izpis sporočila dnevniškega zapisa v srednjem delu dnevnika in pri sestavljanju dnevniških zapisov v oknu za objave.

Kratka oblika datuma / ure v dnevniških zapisih
: Če standardni dnevniški zapisi zavzamejo preveč prostora na zaslonu, uporabite kratek format.

Can double click in log list to compare with previous revision
: If you frequently find yourself comparing revisions in the top pane of the log dialog, you can use this option to allow that action on double click. It is not enabled by default because fetching the diff is often a long process, and many people prefer to avoid the wait after an accidental double click, which is why this option is not enabled by default.

Auto-close
: TortoiseSVN lahko v primeru uspešne operacije samodejno zapre pogovorna okna. Ta nastavitev vam omogoča nastaviti pogoje samodejnega zapiranja. Privzeta (priporočena) nastavitev je Zapri ročno, kar vam omogoča, da pregledate sporočila in preverite, kaj se je zgodilo. Lahko pa se odločite, da vas določene vrste sporočil ne zanimajo in želite, da se takšna okna samodejno zaprejo, če se ne pojavijo kritične napake.

  Izbira Samodejno zapri, če ni spajanj, dodajanj ali brisanj pomeni, da se bo okno napredka zaprlo v primeru enostavnih posodobitev. Če pa pride do spajanja sprememb iz skladišča v delovno kopijo ali če so bile kakšne datoteke dodane ali izbrisane, bo pogovorno okno ostalo odprto. Odprto bo ostalo tudi, če se pri posodobitvi pojavijo spori ali napake.

Izbira Samodejno zapri, če ni sporov še bolj omili pogoje in zapre pogovorna okna tudi v primeru spajanja, dodajanja ali brisanja. Če pa se ob tem pojavijo spori ali napake, pa bo okno vseeno ostalo odprto.

Izbira Samodejno zapri, če ni napak pomeni, da se pogovorno okno vedno zapre, tudi v primeru sporov. Okno ostane odprto le v primeru, če Subversion ne more dokončati operacije. Primer: posodobitev je neuspešna, če strežnik ni dosegljiv, objava ni uspešna, če je delovna kopija zastarela.

Zapri dialoge po lokalnih operacijah
: Local operations like adding files or reverting changes do not need to contact the repository and complete quickly, so the progress dialog is often of little interest. Select this option if you want the progress dialog to close automatically after these operations, unless there are errors.

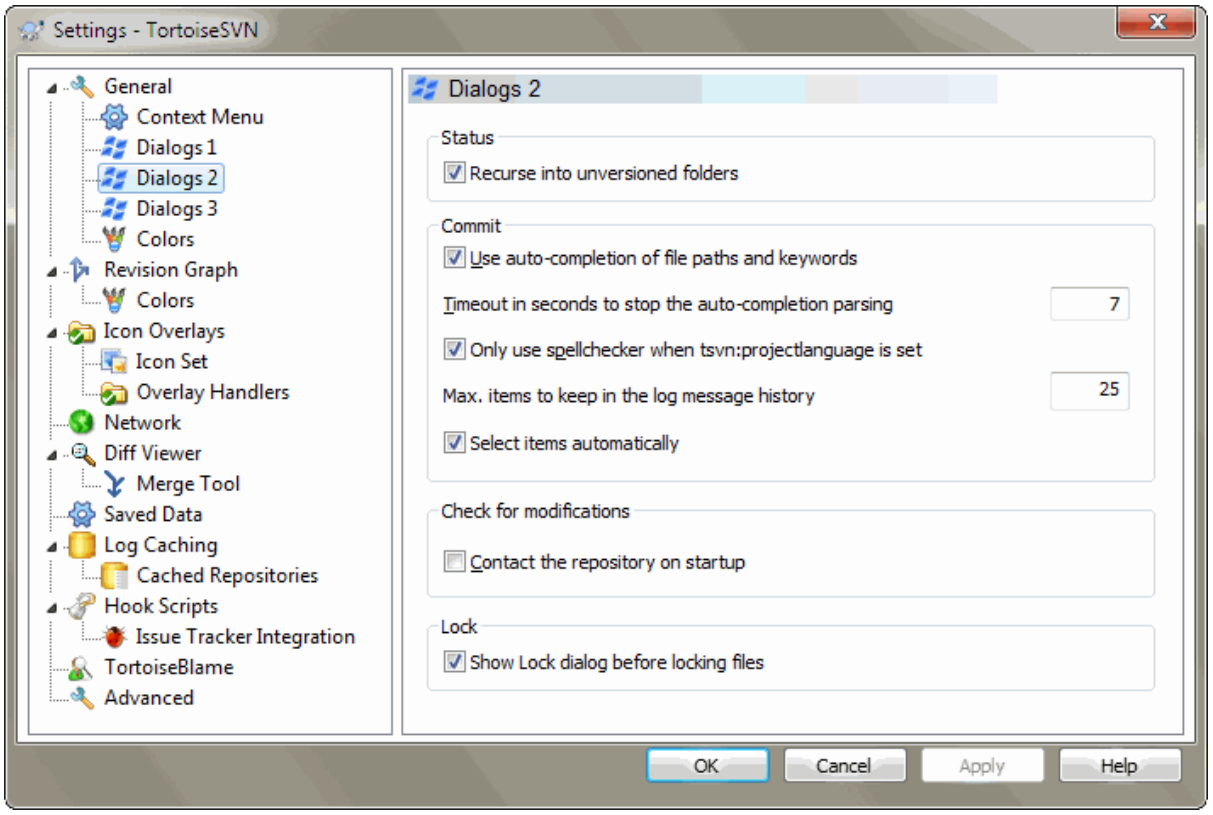
Uporabi koš pri ukazu Povrni
: When you revert local modifications, your changes are discarded. TortoiseSVN gives you an extra safety net by sending the modified file to the recycle bin before bringing back the pristine copy. If you prefer to skip the recycle bin, uncheck this option.

Uporabi URL delovne kopije kot privzeti URL za polje “Iz:‘”
: Po privzetih nastavitvah si okno Spoji zapomni naslov URL, ki je nastavljen v polju Iz:. Nekateri uporabniki opravljajo spajanja iz precej različnih točk v svoji strukture map in jim je lažje, če se v polju pojavi naslov URL trenutne delovne kopije. Naslov URL potem popravijo, da ustreza vzporedni poti na drugi veji.

Privzeta pot za prevzem
: Nastavite lahko privzeto pot za prevzeme. Če imate vse prevzeme na enem mestu, potem je uporabno, da sta pogon in mapa predizpolnjena, tako da dodate na koncu samo ime mape.

Privzeti URL za prevzem
: Lahko pa nastavite privzeti naslov URL za prevzeme. Če pogosto prevzemate podprojekte nekega velike projekta, je uporabno, če imate polje naslov URL predizpolnjeno in mu na koncu dodate samo ime podprojekta.

4.31.1.3. Pogovorna okna 2



Slika 4.75. Okno nastavitev, Pogovorna okna 2

Vstopi rekurzivno v mape brez različic.
: Če je to polje potrjeno (privzeta nastavitev), potem se v oknih Dodaj, Objavi ali Preveri spremembe v primeru map brez različic pojavijo tudi podrejene mape in datoteke brez različic. Če to polje izklopite, potem je prikazana samo vrhnja mapa. Izklop tega polja pripomore k manjši obremenjnosti pogovornih oken. Če v takšnem primeru izberete za dodajanje mapo brez različic, se rekurzivno doda vsa njena vsebina.

  In the Check for Modifications dialog you can opt to see ignored items. If this box is checked then whenever an ignored folder is found, all child items will be shown as well.

Uporabi samodokončanje poti datotek in ključnih besed
: Okno za objave omogoča razčlenjevanje seznama datotek za objavo. Ko vtipkate prve tri črke elementa na seznamu, se pojavi okno za samodokončanje. Pritisnete lahko tipko ENTER in s tem vnesete predlagani niz. Za vklop te zmožnosti potrdite to polje.

Časovni rok v sekundah za ustavitev razčlenjevanja samodokončanja
: Orodje za razčlenjevanje je lahko precej počasno, če dela z velikim številom datotek. Časovni rok prepreči, da bi okno za objave čakalo predolgo časa. Če pri samodokončanju pogrešate določene informacije, podaljšajte časovni rok.

Uporabi črkovalnik le ko je nastavljena lastnost `tsvn:projectlanguage`
: Če ne želite uporabiti črkovalnika za vse objave, potrdite to polje. Črkovalnik bo še vedno omogočen tam, kjer ga zahtevajo projektne lastnosti.

Največje število ohranjenih dnevniških zapisov
: When you type in a log message in the commit dialog, TortoiseSVN stores it for possible re-use later. By default it will keep the last 25 log messages for each repository, but you can customize that number here. If you have many different repositories, you may wish to reduce this to avoid filling your registry.

  Note that this setting applies only to messages that you type in on this computer. It has nothing to do with the log cache.

Elemente izberi samodejno
: Običajno obnašanje okna za objave je, da so vse spremenjene datoteke pod nadzorom izbrane samodejno. Če vam bolj ustreza, da ni izbrana nobena datoteka in boste elemente za objavo izbrali sami, izključite to možnost.

Reopen dialog after commit if items were left uncommitted
: This reopens the commit dialog automatically at the same directory after a successful commit. The dialog is reopened only if there still are items left to commit.

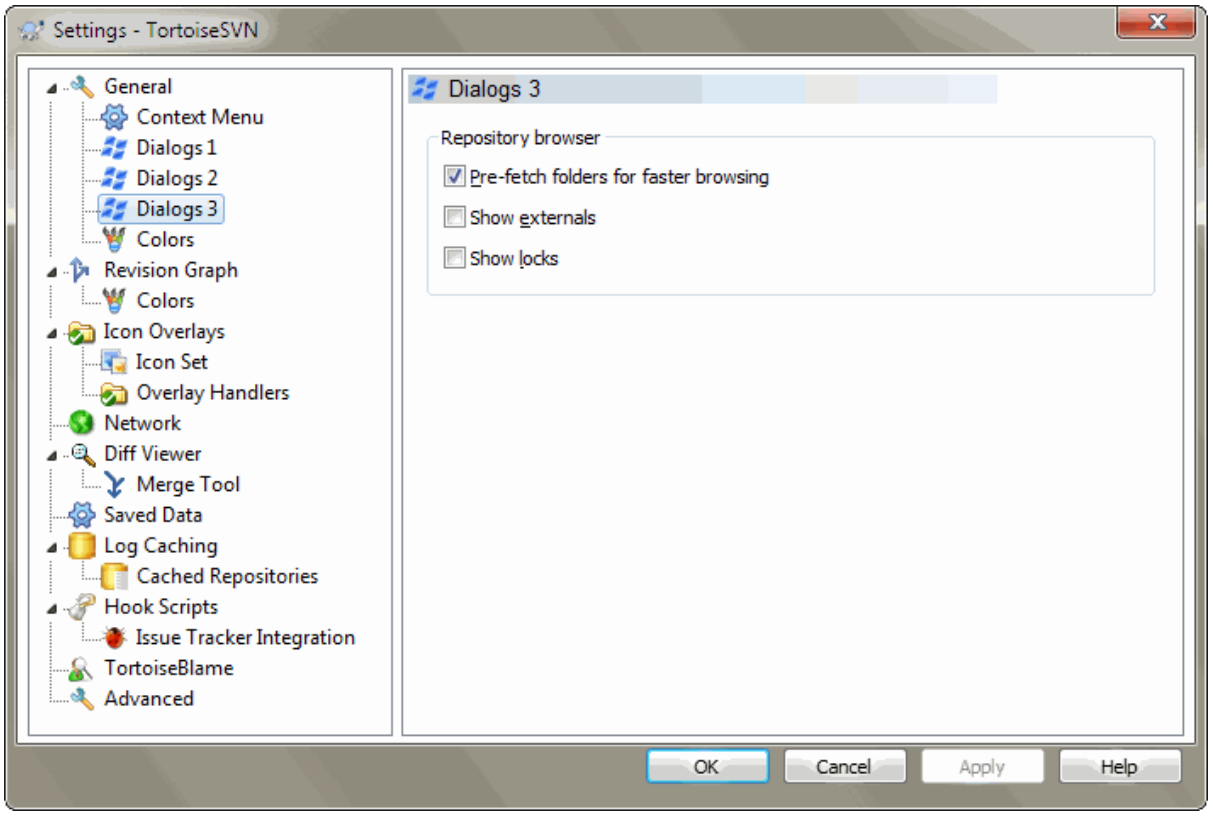
Poveži se s skladiščem ob zagonu
: Okno Preveri spremembe po privzetih nastavitvah preveri le delovno kopijo, skladišče pa preverja le, če kliknete gumb Preveri skladišče. Če želite, da se skladišče vedno samodejno preverja, potrdite to možnost.

Prikaži okno zaklepov pred zaklepanjem datotek
: Pri nekaterih projektih velja pravilo, da se pri zaklepanju ene ali več datotek z ukazom TortoiseSVN → Lock vpiše sporočilo zaklepa z vzrokom zaklepa. Če sporočil zaklepov ne uporabljate, lahko to možnost izključite. S tem preskočite pogovorno kono za vpis sporočila in nemudoma zaklenete datoteke.

  Če uporabite zaklepanje na mapi, se vedno pojavi pogovorno okno za zaklepnaje, ki vam ponudi možnost izbire datotek za zaklepanje.

  Če vaš projekt uporablja lastnost `tsvn:lockmsgminsize`, se bo pogovorno okno zaklepov prikazalo ne glede na te nastavitve, ker projekt *zahteva* zapis zaklepa.

4.31.1.4. TortoiseSVN Dialog Settings 3



Slika 4.76. The Settings Dialog, Dialogs 3 Page

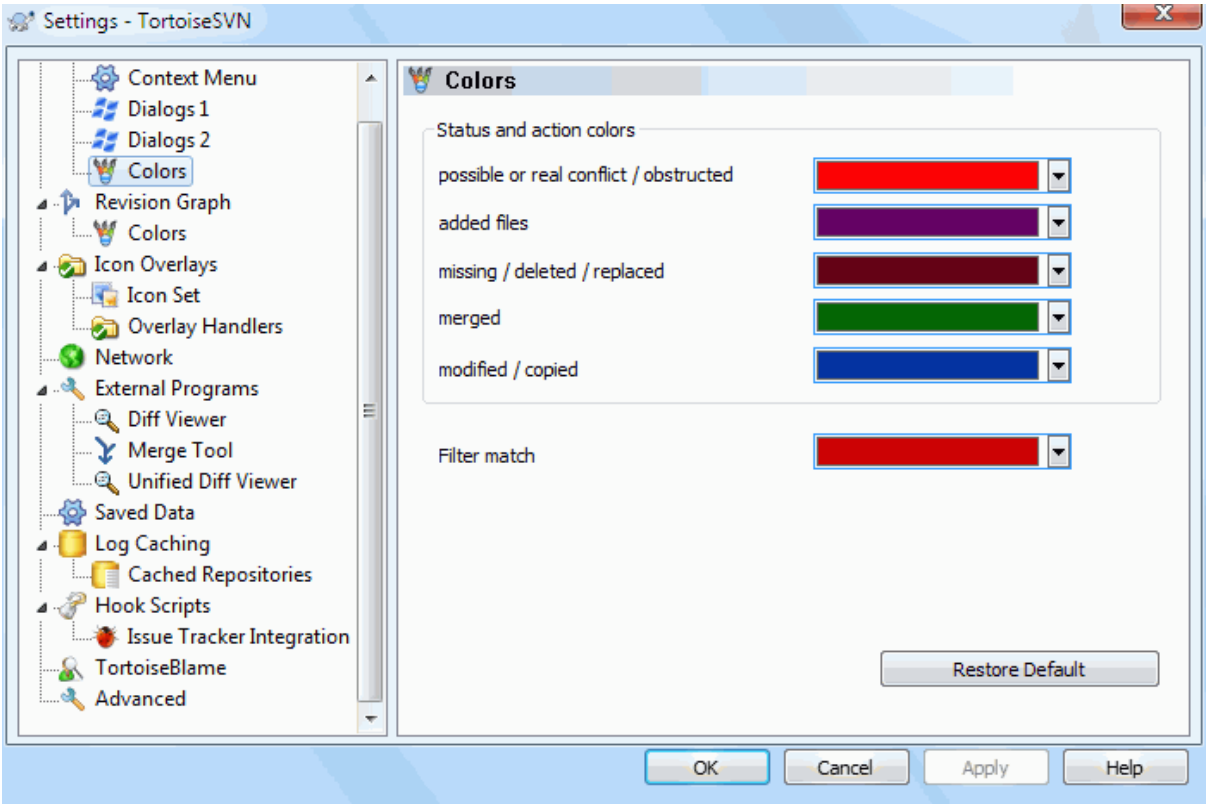
Pre-fetch folders for faster browsing
: If this box is checked (default state), then the repository browser fetches information about shown folders in the background. That way as soon as you browse into one of those folders, the information is already available.

  Some servers however can't handle the multiple requests this causes or when not configured correctly treat so many requests as something bad and start blocking them. In this case you can disable the pre-fetching here.

Pokaži zunanje
: If this box is checked (default state), then the repository browser shows files and folders that are included with the `svn:externals` property as normal files and folders, but with an overlay icon to mark them as from an external source.

  As with the pre-fetch feature explained above, this too can put too much stress on weak servers. In this case you can disable this feature here.

4.31.1.5. Nastavitev barv TortoiseSVN



Slika 4.77. Okno nastavitev, Barve

To pogovorno okno omogoča nastavljanje barve besedil v pogovornih oknih aplikacije TortoiseSVN.

Možen ali resničen spor / ovirano
: Do spora je prišlo med posodobitvijo ali med spajanjem. Posodobitev ovira datoteka/mapa brez različic, ki nosi isto ime kot datoteka/mapa pod nadzorom.

  Ta barva se uporablja tudi za sporočila o napakah v oknu napredka.

Dodane datoteke
: Elementi, dodani v skladišče.

Manjkajoče / izbrisano / zamenjano
: Elementi, izbrisani iz skladišča, ki v delovni kopiji manjkajo, ali element, izbrisan iz delovne kopije in zamenjan z drugo datoteko z istim imenom.

Spojeno
: Spremembe iz skladišča so bile uspešno (brez sporov) spojene v delovno kopijo.

Spremenjeno / kopirano
: Elementi, dodani z zgodovino, ali poti, kopirane v skladišču. Uporablja se tudi v Dnevniku za kopirane elemente.

Izbrisano vozlišče
: Element, ki je bil izbrisan iz skladišča.

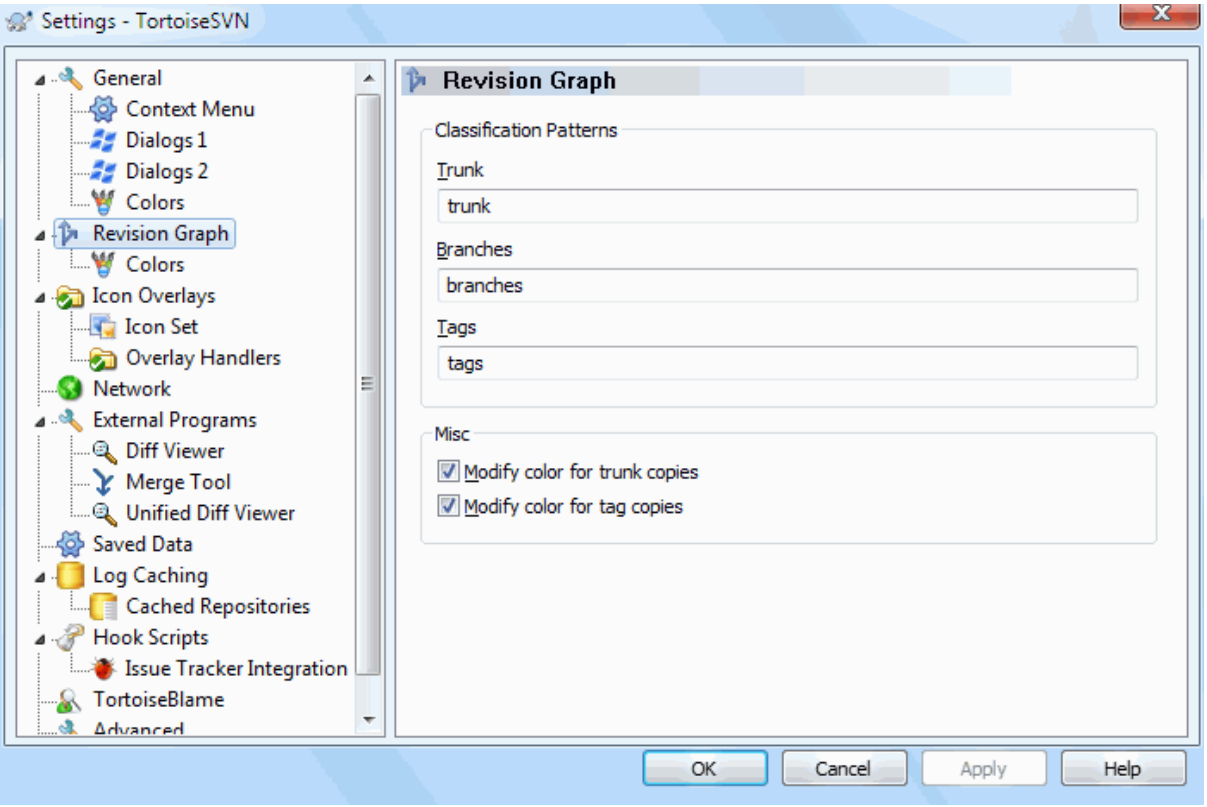
Dodano vozlišče
: Element, ki je bil dodan v skladišče z uporabo ukazov dodaj, kopiraj ali premakni.

Preimenovano vozlišče
: Element, ki je bil preimenovan znotraj skladišča.

Zamenjano vozlišče
: Izviren element je bil izbrisan. Nadomesti ga nov element z enakim imenom.

Filter poti
: When using filtering in the log dialog, search terms are highlighted in the results using this colour.

4.31.2. Revision Graph Settings



Slika 4.78. The Settings Dialog, Revision Graph Page

Vzorec razvrstitve
: The revision graph attempts to show a clearer picture of your repository structure by distinguishing between trunk, branches and tags. As there is no such classification built into Subversion, this information is extracted from the path names. The default settings assume that you use the conventional English names as suggested in the Subversion documentation, but of course your usage may vary.

  Specify the patterns used to recognise these paths in the three boxes provided. The patterns will be matched case-insensitively, but you must specify them in lower case. Wild cards `*` and `?` will work as usual, and you can use `;` to separate multiple patterns. Do not include any extra white space as it will be included in the matching specification.

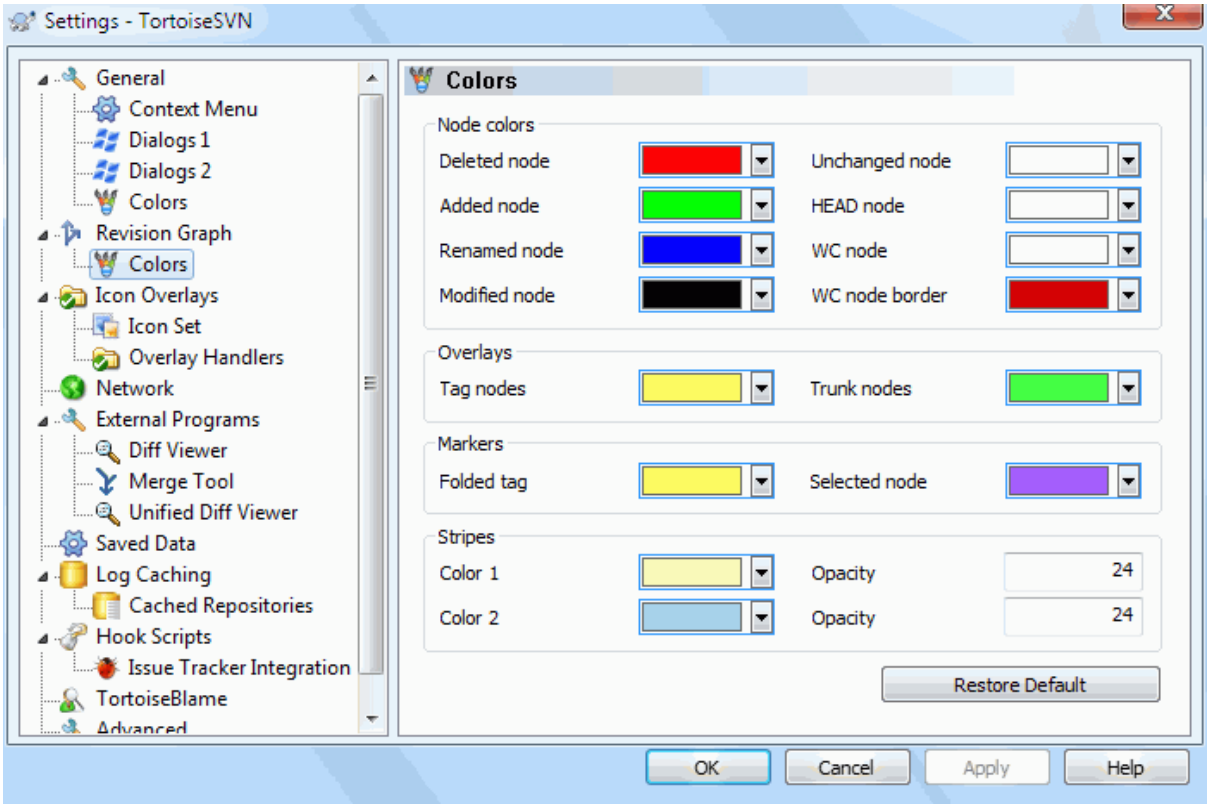
Commit tag detection

Please note that these patterns are also used to detect commits to a tag, not just for the revision graph.

Modify Colors
: Colors are used in the revision graph to indicate the node type, i.e. whether a node is added, deleted, renamed. In order to help pick out node classifications, you can allow the revision graph to blend colors to give an

indication of both node type and classification. If the box is checked, blending is used. If the box is unchecked, color is used to indicate node type only. Use the color selection dialog to allocate the specific colors used.

4.31.2.1. Revision Graph Colors



Slika 4.79. The Settings Dialog, Revision Graph Colors Page

This page allows you to configure the colors used. Note that the color specified here is the solid color. Most nodes are colored using a blend of the node type color, the background color and optionally the classification color.

Deleted Node
: Items which have been deleted and not copied anywhere else in the same revision.

Added Node
: Items newly added, or copied (add with history).

Renamed Node
: Items deleted from one location and added in another in the same revision.

Modified Node
: Simple modifications without any add or delete.

Unchanged Node
: May be used to show the revision used as the source of a copy, even when no change (to the item being graphed) took place in that revision.

Vzolišče HEAD
: Current HEAD revision in the repository.

WC Node
: If you opt to show an extra node for your modified working copy, attached to its last-commit revision on the graph, use this color.

WC Node Border
: If you opt to show whether the working copy is modified, use this color border on the WC node when modifications are found.

Tag Nodes
: Nodes classified as tags may be blended with this color.

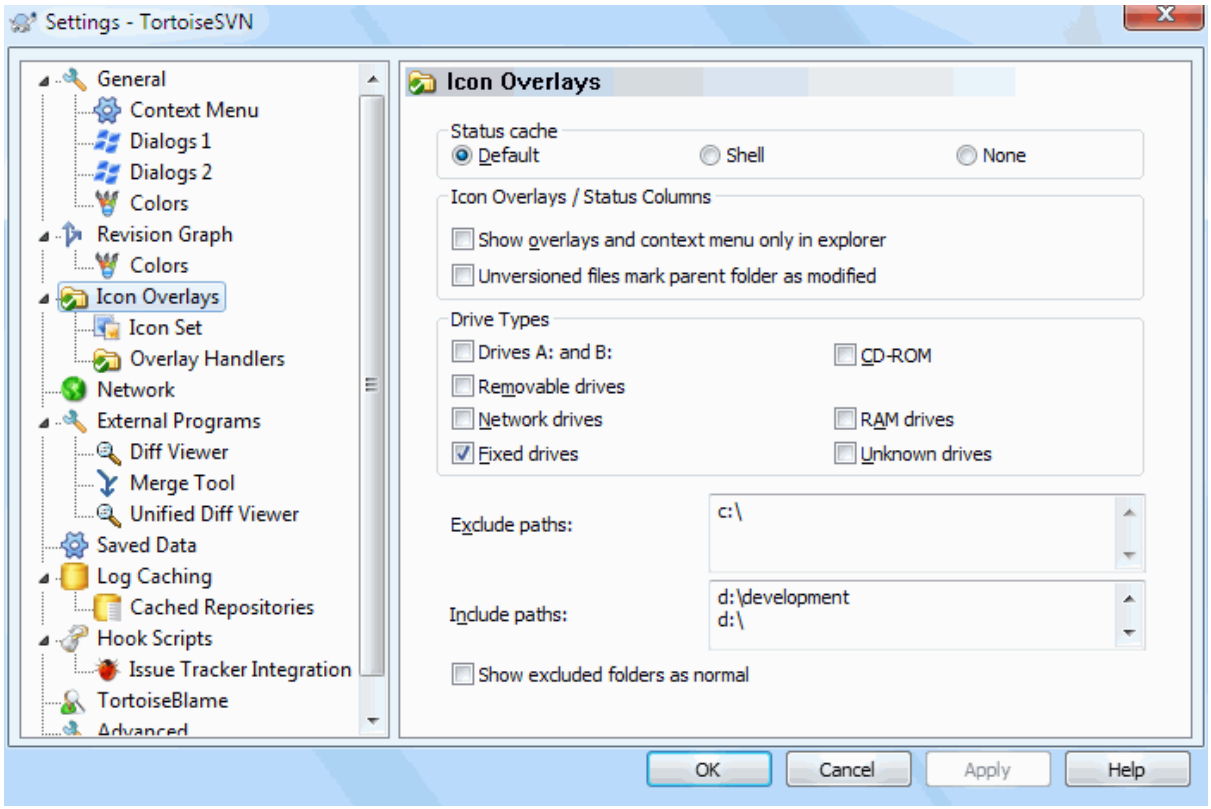
Trunk Nodes
: Nodes classified as trunk may be blended with this color.

Folded Tag Markers
: If you use tag folding to save space, tags are marked on the copy source using a block in this color.

Selected Node Markers
: When you left click on a node to select it, the marker used to indicate selection is a block in this color.

Proge
: These colors are used when the graph is split into sub-trees and the background is colored in alternating stripes to help pick out the separate trees.

4.31.3. Nastavitve prekrivnih ikon



Slika 4.80. Okno nastavitev, Izbor ikon

Ta stran omogoča izbiro elementov, za katere TortoiseSVN prikaže prekrivne ikone.

Ker pridobivanje informacij o stanju delovne kopije traja kar nekaj časa, TortoiseSVN uporablja predpomnilnik, tako da prikazovanje prekrivnih ikon poteka bolj gladko. Glede na vaš sistem in velikost delovne kopije izberite ustrezen tip predpomnilnika:

Prizeto
: Vse informacije o stanju pridobiva poseben proces (`TSVNCache.exe`). Ta preverja spremembe na vseh pogonih in ponovno pridobi informacijo o stanju, če se datoteke znoraj delovne kopije spremenijo. Proces

teče z najmanjšo možno prioriteto izvajanja, tako da ne upočasni ostalih aplikacij. To pa pomeni tudi, da se informacije o procesu ne prodobivajo v *realnem času*, zato se lahko sprememba prekrivne ikone zgodi šele čez nekaj sekund.

Prednost: prekrivne ikone prikazujejo status rekurzivno: če se spremeni datoteka nekje globoko v hierarhiji map, se prekrivne ikone spremenijo vsem mapam do korenske mape delovne kopije. Ker lahko proces pošilja obvestila o spremembah ukazni lupini, se običajno spremenijo tudi prekrivne ikone map na levi strani Raziskovalca (drevo strukture map).

Slabost: proces teče neprestano, tudi če ne delate na projektih. Uporablja pa tudi 10 do 50 MB pomnilnika (RAM), odvisno od števila in velikosti delovnih kopij.

Lupina
: Polnjenje predpomnilnika se dogaja znotraj knjižnice za razširitev lupine, vendar zgolj za trenutno vidno mapo. Vsakič, ko se premaknete na drugo mapo, se informacije o stanju ponovno pridobijo.

  Prednost: potrebuje zelo malo pomnilnika (približno 1MB), stanje pa lahko prikazuje v *realnem času*.

  Slabost: Ker se v predpomnilnik shranijo le informacije o eni mapi, se prekrivne ikone ne prikazujejo rekurzivno. Pri velikih delovnih kopijah lahko prikaz mape traja dalj časa kot pri privzetih nastavitvah. Stolpec, ki prikazuje tip MIME, ni na voljo.

Brez predpomnilnika
: Pri tej nastavitvi TortoiseSVN ne pridobiva stanja delovne kopije v Raziskovalcu. Datoteke nimajo prekrivnih ikon, mape pa imajo prekrivno ikono 'običajno', če so pod nadzorom različic. Druge prekrivne ikone se ne prikazujejo, prav tako niso na voljo dodatni stolpci z informacijami.

  Prednost: ne uporablja nobenega dodatnega spomina in ne upočasni delovanja Raziskovalca med brskanjem.

  Slabost: informacija o stanju datotek in map ni prikazana v Raziskovalcu. Če želite preveriti, ali je vaša delovna kopija spremenjena, morate uporabiti okno “Preveri spremembe”.

Po privzetih nastavitvah se prekrivne ikone in kontekstni meni pojavijo v vseh oknih Open/Save (Odpri/Shrani) in v Raziskovalcu. Če želite, da se pojavijo *samo* v Raziskovalcu, potrdite polje Pokaži prekrivanje ikon in kontekstni meni samo v raziskovalcu.

You can force the status cache to *None* for elevated processes by checking the Disable status cache for elevated processes box. This is useful if you want to prevent another `TSVNCache.exe` process getting created with elevated privileges.

Če izberete privzeto možnost, lahko izberete tudi možnost označevanja map kot spremenjenih, če vsebujejo elemente brez različic. To je uporabno, saj vas opozarja, da ste ustvarili nove datoteke, ki še niso dodane v skladišče. Ta možnost je na voljo le, če uporabljate *privzet* način predpomnilnika stanja (spodaj).

If you have files in the `ignore-on-commit` changelist, you can chose to make those files not propagate their status to the parent folder. That way if only files in that changelist are modified, the parent folder still shows the unmodified overlay icon.

Naslednja skupina omogoča nastavite vrste elementov, za katere se prikažejo prekrivne ikone. Po privzetih nastavitvah so izbrani le trdi diski. Seveda lahko prikaz prekrivnih ikon popolnoma izključite, toda to sploh ni zabavno, kaj ne?

Omrežni pogoni so zelo počani, zato po privzetih nastavitvah prekrivne ikone delovnih kopij na omrežnih pogonih niso prikazane.

Pogoni USB so poseben primer, saj je njihov tip definiran s pogonom samim. Tako se nekateri pojavijo kot stalni pogoni, nekateri pa kot odstanjivi.

The Exclude Paths are used to tell TortoiseSVN those paths for which it should *not* show icon overlays and status columns. This is useful if you have some very big working copies containing only libraries which you won't change at all and therefore don't need the overlays, or if you only want TortoiseSVN to look in specific folders.

Any path you specify here is assumed to apply recursively, so none of the child folders will show overlays either. If you want to exclude *only* the named folder, append `?` after the path.

Enako velja tudi za nastavitev Vključi poti. Za te poti se prekrivne ikone prikazujejo vedno, tudi če so sicer izklopljene za ta tip pogona ali preko zgornjih nastavitev.

Users sometimes ask how these three settings interact. For any given path check the include and exclude lists, seeking upwards through the directory structure until a match is found. When the first match is found, obey that include or exclude rule. If there is a conflict, a single directory spec takes precedence over a recursive spec, then inclusion takes precedence over exclusion.

An example will help here:

```
Exclude:
C:
C:\develop\?
C:\develop\tsvn\obj
C:\develop\tsvn\bin

Include:
C:\develop
```

These settings disable icon overlays for the C: drive, except for `c:\develop`. All projects below that directory will show overlays, except the `c:\develop` folder itself, which is specifically ignored. The high-churn binary folders are also excluded.

Te poti uporablja tudi TSVNCache.exe, da si omeji pregledovanje. Če želite, da pregleda samo določene mape, onemogočite vse tipe pogonov in vključite le mape, ki jih želite pregledati.

Izločite pogone `SUBST`

It is often convenient to use a `SUBST` drive to access your working copies, e.g. using the command

```
subst T: C:\TortoiseSVN\trunk\doc
```

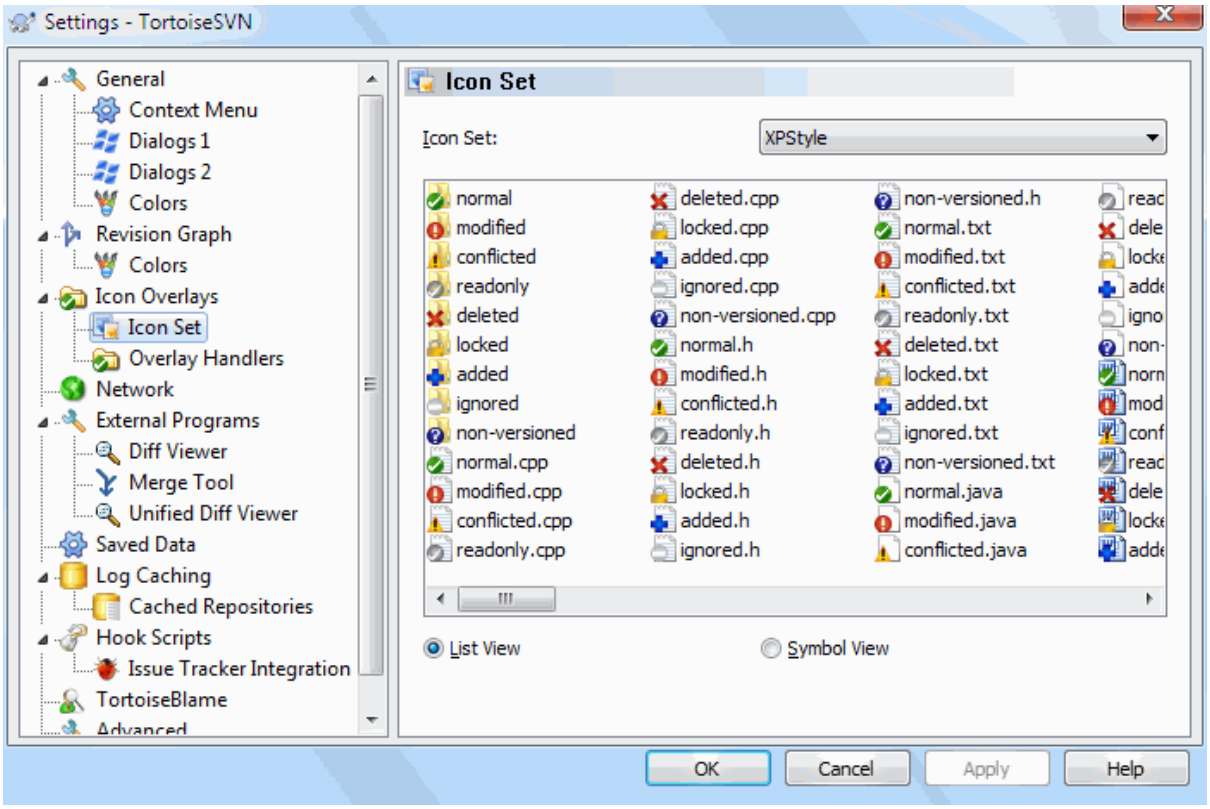
However this can cause the overlays not to update, as `TSVNCache` will only receive one notification when a file changes, and that is normally for the original path. This means that your overlays on the `subst` path may never be updated.

Da se izognemu temu problemu, izvirno pot dodamo na seznam poti, ki ji TortoiseSVN prezre pri posodabljanju prekrivnih ikon. Na ta način se bodo posodabljale ikone na pogonu, narejenem z ukazom `subst`.

Sometimes you will exclude areas that contain working copies, which saves TSVNCache from scanning and monitoring for changes, but you still want a visual indication that a folder contains a working copy. The Show excluded root folders as 'normal' checkbox allows you to do this. With this option, working copy root folders in any excluded area (drive type not checked, or specifically excluded) will show up as normal and up-to-date, with a green check mark. This reminds you that you are looking at a working copy, even though the folder overlays may not be correct. Files do not get an overlay at all. Note that the context menus still work, even though the overlays are not shown.

Izjema sta pogona `A:` in `B:`, ki se nikoli ne upoštevata pri možnosti Prikaži izločene mape kot 'navadne'. V takem primeru bi namreč sistem Windows pregledal omenjena pogona, tudi če ju dejansko nimate, kar bi za nekaj sekund upočasnilo zaganjanje Raziskovalca.

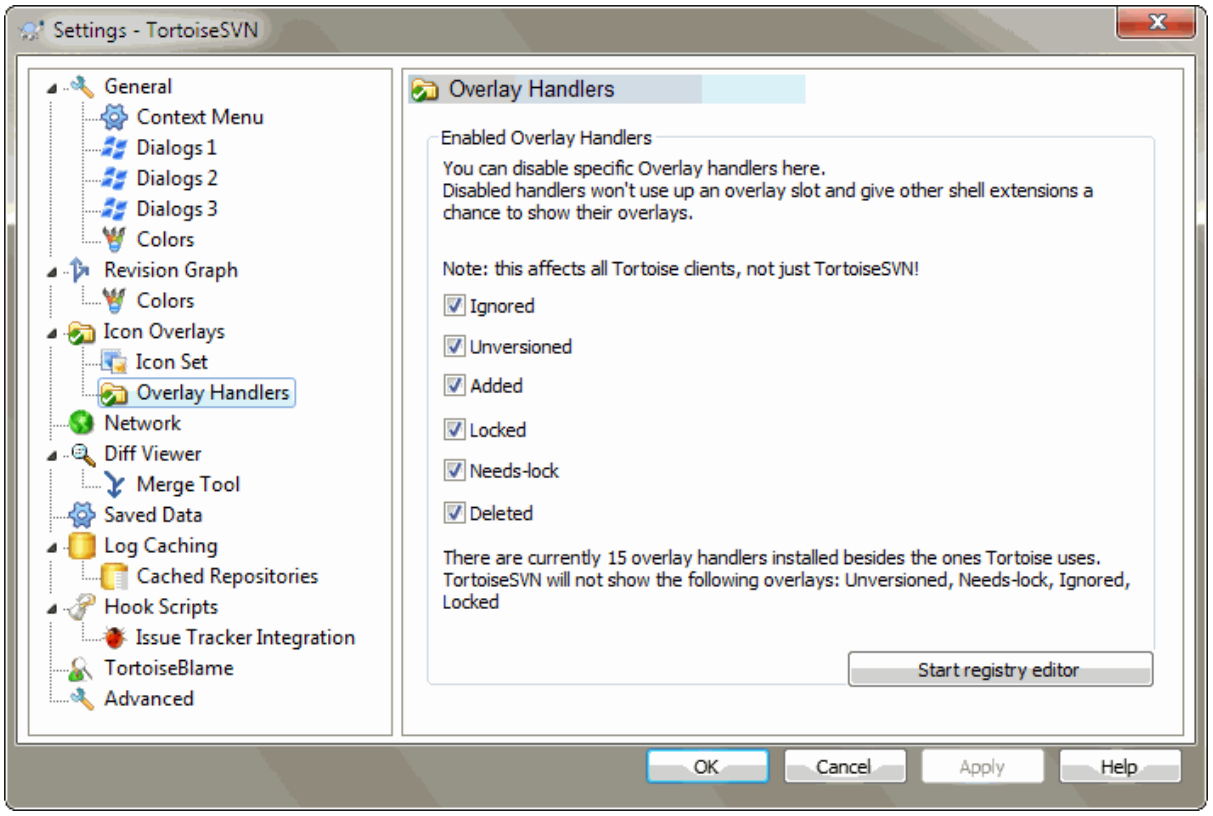
4.31.3.1. Izbira nabora ikon



Slika 4.81. Okno nastavitev, Izbor ikon

Izberete si lahko tak nabor ikon, ki vam najbolj ustreza. Da bi uveljavili spremembe, boste morali mogoče ponovno zagnati računalnik.

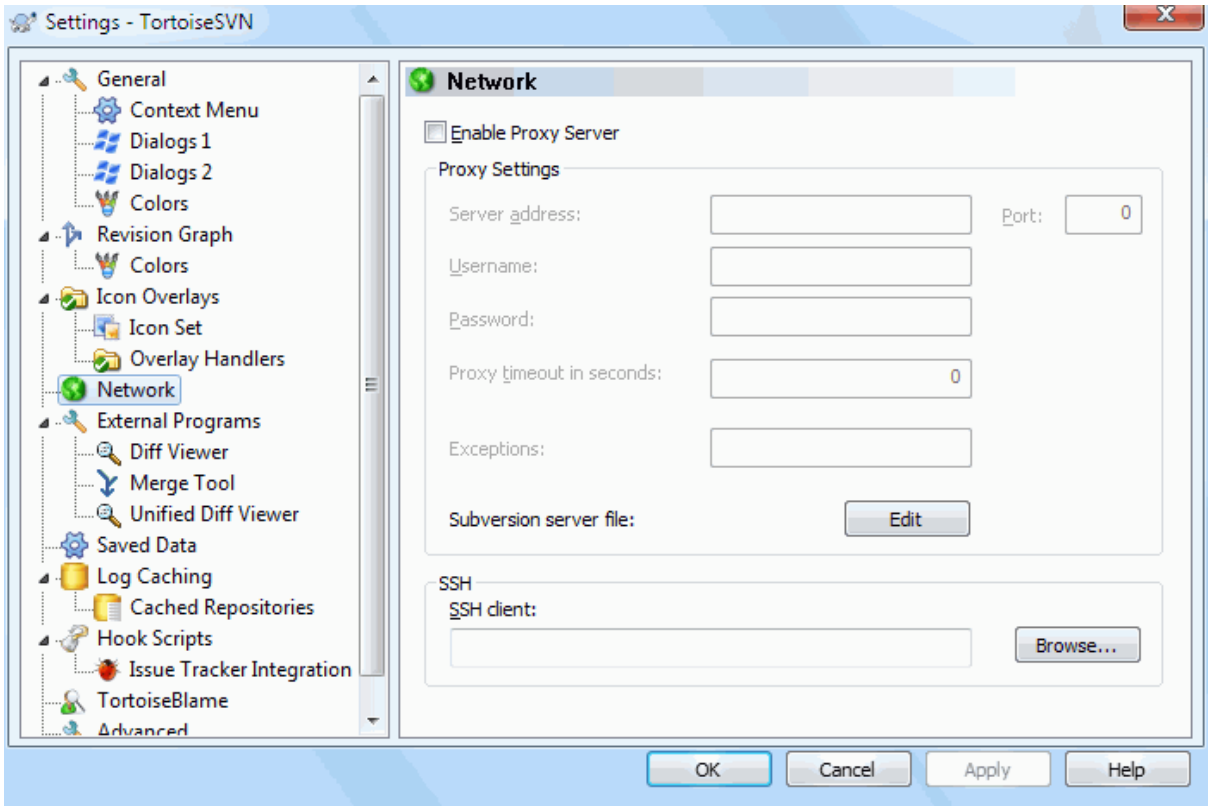
4.31.3.2. Omogočeni načini prekrivanja ikon



Slika 4.82. The Settings Dialog, Icon Handlers Page

Because the number of overlays available is severely restricted, you can choose to disable some handlers to ensure that the ones you want will be loaded. Because TortoiseSVN uses the common TortoiseOverlays component which is shared with other Tortoise clients (e.g. TortoiseCVS, TortoiseHg) this setting will affect those clients too.

4.31.4. Nastavitve omrežja



Slika 4.83. Okno nastavitev, Omrežje

Tukaj lahko nastavite posredniški strežnik (proxy). Mogoče ga potrebujete, da lahko delate preko požarnega zidu v podjetju.

If you need to set up per-repository proxy settings, you will need to use the Subversion `servers` file to configure this. Use Edit to get there directly. Consult the *Runtime Configuration Area* [http://svnbook.red-bean.com/en/1.8/svn.advanced.confarea.html] for details on how to use this file.

Določite lahko aplikacijo, ki naj jo TortoiseSVN uporabi za vzpostavitev varne povezave do skladišča svn+ssh. Priporočamo uporabo programa TortoisePlink.exe. To je posebna različica priljubljenega programa Plink in je del paketa TortoiseSVN. Prevedena je kot aplikacija brez okna, tako da se ob avtentikaciji ne pojavi okno DOS.

You must specify the full path to the executable. For TortoisePlink.exe this is the standard TortoiseSVN bin directory. Use the Browse button to help locate it. Note that if the path contains spaces, you must enclose it in quotes, e.g.

```
"C:\Program Files\TortoiseSVN\bin\TortoisePlink.exe"
```

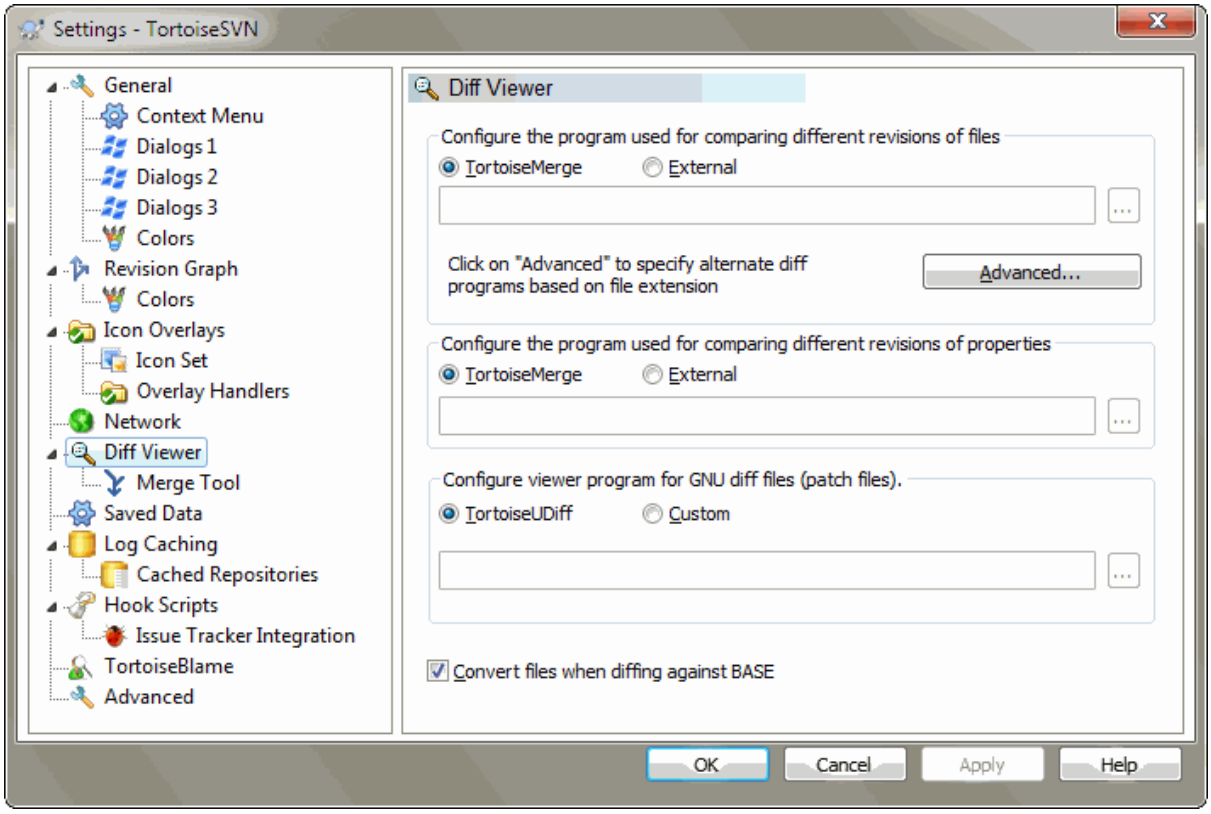
Stranski učinek aplikacije brez okna je, da ta ne more izpisati sporočila o napaki. Če avtentikacija ne uspe, dobite sporočilo tipa “Ne morem pisati na standardni izhod”. Zato priporočamo, da najprej uporabite standarno različico programa Plink. Ko vse deluje, pa jo zamenjate s programom TortoisePlink. Uporabite popolnoma enake parametre.

TortoisePlink does not have any documentation of its own because it is just a minor variant of Plink. Find out about command line parameters from the *PuTTY website* [https://www.chiark.greenend.org.uk/~sgtatham/putty/].

Da vam ne bi bilo potrebno vsakokrat vpisovati uporabniškega imena in gesla, lahko uporabite orodje za shranjevane gesel, naprimer Pageant, ki je na voljo na domači strani PuTTY.

Finally, setting up SSH on server and clients is a non-trivial process which is beyond the scope of this help file. However, you can find a guide in the TortoiseSVN FAQ listed under *Subversion/TortoiseSVN SSH How-To* [https://tortoisesvn.net/ssh_howto.html].

4.31.5. Nastaviteve zunanjih programov



Slika 4.84. Okno nastavitev, ogledovalnik razlik

Tukaj lahko nastavite programe za razlikovanje/spajanje, ki naj jih TortoiseSVN uporabi. Privzeta nastavitev je uporaba programa TortoiseMerge, ki se namesti skupaj s TortoiseSVN.

Preberite Razdelek 4.11.6, “Zunanja orodja za razlikovanje/spajanje”, kjer so navedena nekatera zunanja orodja za razlikovanje/spajanje, ki jih uporabniki uporabljajo skupaj s sistemom TortoiseSVN.

4.31.5.1. Pregledovalnik razlik

Za primerjanje razlik med različnimi revizijami datotek lahko uporabljate zunanji program za razlikovanje. Zunanji program mora v ukazni vrstici dobiti imena datotek in parametre ukazne vrstice. TortoiseSVN uporablja parametre za zamenjavo s predpono `%`. Ko naleti na parameter, ga zamenja z ustrezno vrednostjo. Vrstni red parametrov je odvisen od programa za razlikovanje, ki ga uporabljate.

%base
: Izvirna datoteka brez vaših sprememb

%bname
: Naslov okna osnovne datoteke

%nqbname
: The window title for the base file, without quotes

%mine
: Vaša lastna datoteka z vašimi spremembami

%yname
Naslov okna za vašo datoteko

%nqyname
The window title for your file, without quotes

%burl
The URL of the original file, if available

%nqburl
The URL of the original file, if available, without quotes

%yurl
The URL of the second file, if available

%nqyurl
The URL of the second file, if available, without quotes

%brev
The revision of the original file, if available

%nqbrev
The revision of the original file, if available, without quotes

%yrev
The revision of the second file, if available

%nqyrev
The revision of the second file, if available, without quotes

%peg
The peg revision, if available

%nqpeg
The peg revision, if available, without quotes

%fname
The name of the file. This is an empty string if two different files are diffed instead of two states of the same file.

%nqfname
The name of the file, without quotes

The window titles are not pure filenames. TortoiseSVN treats that as a name to display and creates the names accordingly. So e.g. if you're doing a diff from a file in revision 123 with a file in your working copy, the names will be `filename : revision 123` and `filename : working copy`.

For example, with ExamDiff Pro:

```
C:\Path-To\ExamDiff.exe %base %mine --left_display_name:%bname
                                    --right_display_name:%yname
```

or with KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine --L1 %bname --L2 %yname
```

or with WinMerge:

```
C:\Path-To\WinMerge.exe -e -ub -dl %bname -dr %yname %base %mine
```

or with Araxis:

```
C:\Path-To\compare.exe /max /wait /title1:%bname /title2:%yname
                       %base %mine
```

or with UltraCompare:

```
C:\Path-To\uc.exe %base %mine -title1 %bname -title2 %yname
```

or with DiffMerge:

```
C:\Path-To\DiffMerge.exe -nosplash -t1=%bname -t2=%yname %base %mine
```

Če uporabljate lastnost `svn:keywords` za razširitev ključnih besed, posebej če uporabite *revizijo* datoteke, se med datotekami pojavi razlike zgolj zaradi različne vrednosti ključnih besed. Podobno je pri uporabi lastnosti `svn:eol-style = native`: osnovna datoteka (BASE) bo uporabljala `LF` za zaključek vrstice, vaša datoteka pa `CR-LF`. TortoiseSVN običajno te razlike skrije, tako da pred primerjanjem tudi v osnovni datoteki razširi ključne besede in spremeni zaključke vrstic. To pa lahko traja precej časa, če delamo z dolgimi datotekami. Če je polje Pri razlikovanju glede na BASE pretvori datoteke izklopljeno, bo TortoiseSVN preskočil predprocesiranje datotek.

Lahko pa določite tudi drugo orodje za razlikovanje lastnosti v sistemu Subversion. Ker so lastnosti ponavadi enostavna kratka besedila, vam bo mogoče bolj ustrezal enostavnejši pregledovalnik.

Če ste nastavili dodatna orodja za razlikovanje, lahko iz kontekstnega menija izbere TortoiseMerge *in* ostala orodja. Kontekstni meni → Razlikuj uporabi osnovno orodje za razlikovanje, **Shift**+Kontekstni meni → Razlikuj pa dodatnega.

At the bottom of the dialog you can configure a viewer program for unified-diff files (patch files). No parameters are required. The Default setting is to use TortoiseUDiff which is installed alongside TortoiseSVN, and colour-codes the added and removed lines.

Since Unified Diff is just a text format, you can use your favourite text editor if you prefer.

4.31.5.2. Orodje za spajanje

Zunanji program za spajanje, uporabljen za reševaje spornih datotek. Zamenjava parametrov se uporablja prav tako kot pri programu za razlikovanje.

%base
: izvirna datoteka brez vaših sprememb ali sprememb ostalih

%bname
: Naslov okna osnovne datoteke

%nqbname
: The window title for the base file, without quotes

%mine
: vaša datoteka z vašimi spremembami

%yname
: Naslov okna za vašo datoteko

%nqyname
: The window title for your file, without quotes

%theirs
: datoteka, kot se nahaja v skladišču

%tname
: naslov okna za datoteko v skladišču

%nqtname
: The window title for the file in the repository, without quotes

%merged
: sporna datoteka, rezultat spajanja

%mname
: naslov okna za spojeno datoteko

%nqmname
: The window title for the merged file, without quotes

%fname
: The name of the conflicted file

%nqfname
: The name of the conflicted file, without quotes

For example, with Perforce Merge:

```
C:\Path-To\P4Merge.exe %base %theirs %mine %merged
```

or with KDiff3:

```
C:\Path-To\kdiff3.exe %base %mine %theirs -o %merged
                      --L1 %bname --L2 %yname --L3 %tname
```

or with Araxis:

```
C:\Path-To\compare.exe /max /wait /3 /title1:%tname /title2:%bname
                       /title3:%yname %theirs %base %mine %merged /a2
```

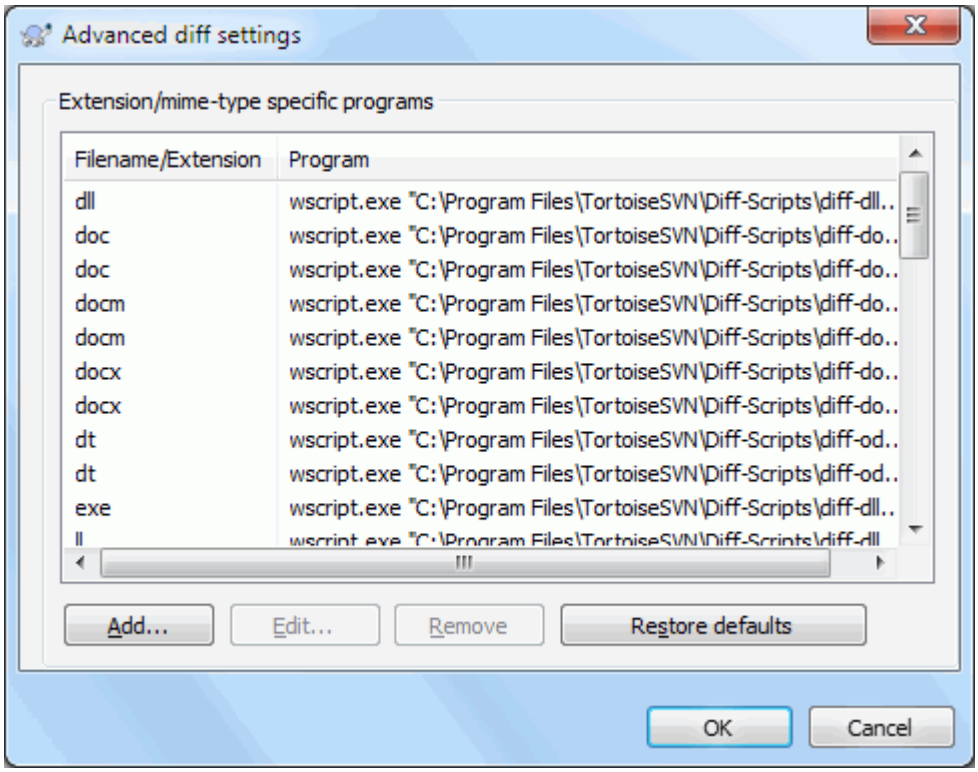
or with WinMerge (2.8 or later):

```
C:\Path-To\WinMerge.exe %merged
```

or with DiffMerge:

```
C:\Path-To\DiffMerge.exe -caption=%mname -result=%merged -merge
      -nosplash -t1=%yname -t2=%bname -t3=%tname %mine %base %theirs
```

4.31.5.3. Napredne nastavitve za razlikovanje/spajanje

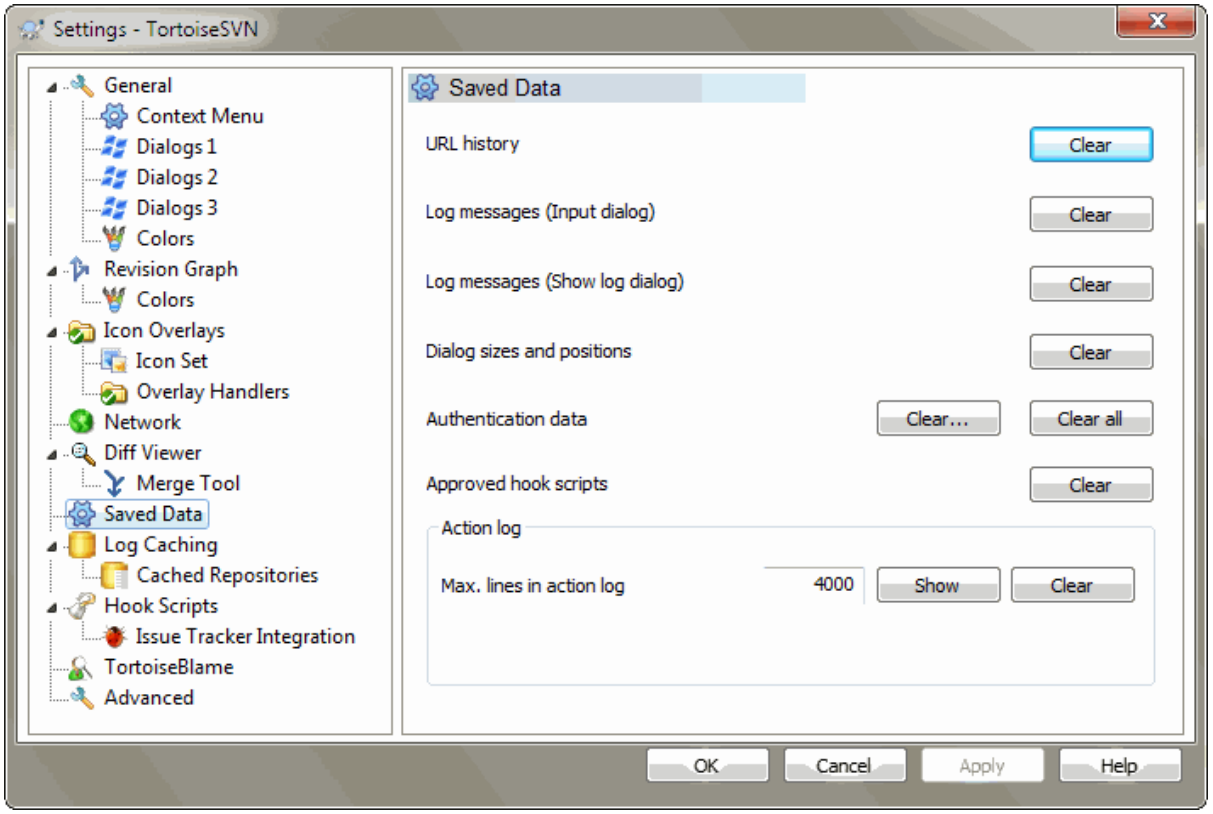


Slika 4.85. Okno nastavitev, napredne nastavitve razlikovanja/spajanja

V naprednih nastavitvah lahko nastavite različna orodja za razlikovanje in spajanje glede na končnico datoteke. Primer: Photoshop nastavite kot program za “razlikovanje” datotek `.jpg` :-) Tudi lastnost `svn:mime-type` lahko asociirate s programom za razlikovanje ali spajanje.

Za povezavo orodij s končnico datotek morate navesti končnico. Uporabite `.bmp` za datoteke bitmap sistema Windows. Za povezavo orodij z lastnostjo `svn:mime-type` nastavite tip MIME, vključno s poševnico, n.pr. `text/xml`.

4.31.6. Shranjeni podatki



Slika 4.86. Okno nastavitev, Shranjeni podatki

Za večjo uporabnost TortoiseSVN shrani veliko nastavitev, ki jih uporabljate, in si zapomni vaše zadnje operacije. Če želite, lahko tu te podatke izbrišete.

Zgodovina naslovov URL
: Ko naredite prevzem delovne kopije, spojite spremembe ali uporabite brskalnik po skladišču, vam TortoiseSVN ponudi shranjene zadnje uporabljene naslove URL v spustnem meniju. Včasih je seznam poln starih naslovov in ga je zato koristno počistiti.

  Če želite iz spustnega polja izbrisati posamezne elemente, lahko to storite kar na polju samem. Kliknite na puščico za spust spustnega polja, premaknite miško na element, ki ga želite odstraniti in stisnite **Shift+Del**.

Dnevniška sporočila (vnosno okno)
: TortoiseSVN shrani zadnjih nekaj sporočil dnevniških zapisov, ki jih vnesete in sicer za vsako skladišče posebej. Če uporabljate veliko skladišč, postane ta seznam precej dolg.

Dnevniška sporočila (okno Prikaži dnevnik)
: TortoiseSVN si ob prikazu okna sporočila dnevniških zapisov shrani v predpomnilnik, tako da naslednjič, ko jih mora prikazati, prihrani čas. Če nekdo dnevniški zapis popravi, vi pa ta zapis že imate v predpomnilniku, te spremembe ne boste videli, dokler ne izbrišete predpomnilnika. Predpomnenje dnevniških zapisov se vključi v zavihku Predpomnilnik dnevnika.

Velikosti in položaji pogovornih oken
: Veliko oken si zapomni velikost in položaj na zaslonu ob zadnji uporabi.

Podatki avtentikacije
: Ko uporabljate avtentikacijo s strežnikom Subversion, se uporabniško ime in geslo shranita v krajevni datoteki in vam ju tako ni potrebno vedno znova vnašati. Te podatke boste morda želeli izbrisati - zaradi varnostnih razlogov ali pa zato, ker želite do skladišča dostopati z drugim uporabniškim imenom ... ali Janez ve, da uporabljate njegov PC?

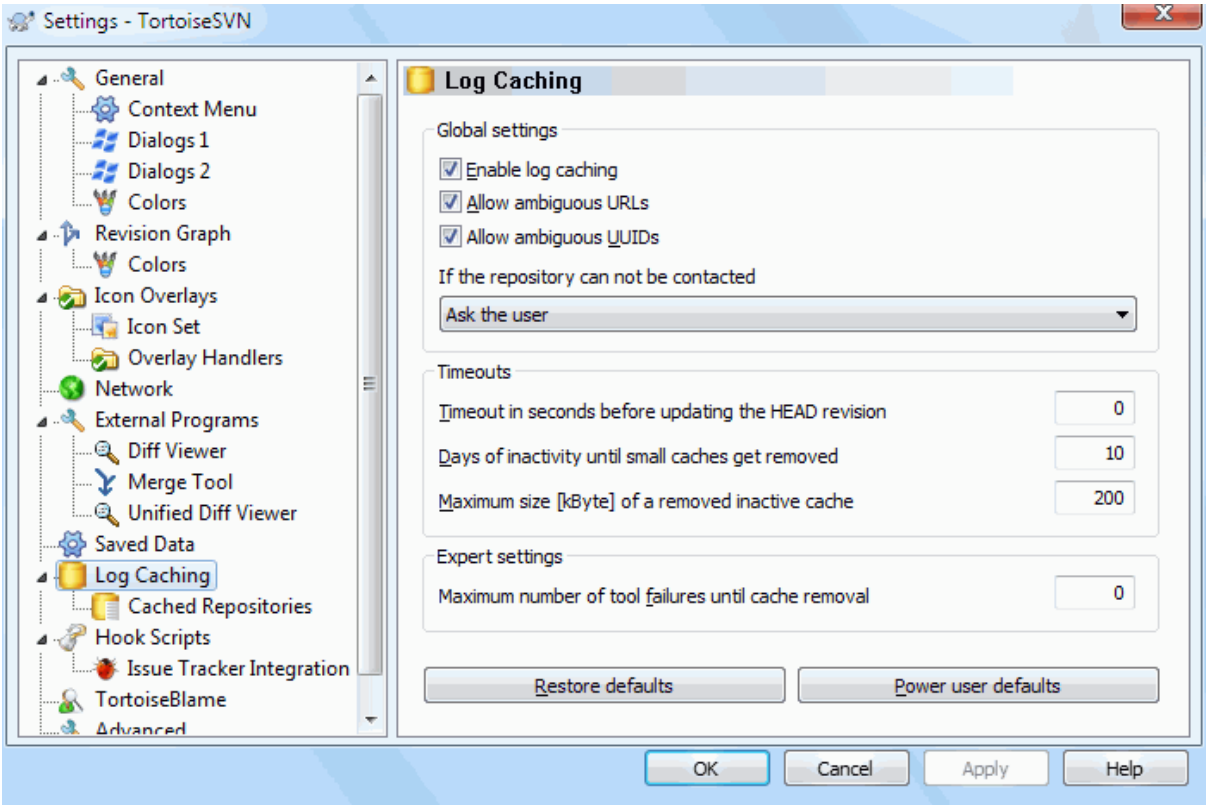
If you want to clear authentication data for one particular server only, use the Clear... instead of the Clear all button.

Zapisi dejanj
: TortoiseSVN hrani zapise vsega, kaj se izpisuje v pogovorna okna napredka. To je uporabno, če želite, na primer, preveriti, kaj se je zgodilo pri zadnji posodobitvi delovne kopije.

  Datoteka zapisa je omejena na določeno dolžino. Ko to dolžino preseže, se najstarejši zapisi odstranijo. Po privzetih nastavitvah se ohrani 4000 vrstic, vendar lahko to številko spremenite.

  Tukaj lahko pogledate vsebino zapisov, prav tako pa lahko vsebino počistite.

4.31.7. Predpomnenje dnevnika



Slika 4.87. Okno nastavitev, Predpomnilnik dnevnika

To okno omogoča nastavitev predpomnenja. Predpomnenje shrani krajevno kopijo dnevniških zapisov in sprememb poti, tako da se prihrani dolgotrajno prenašanje teh podatkov s strežnika. Uporaba predpomnenja drastično skrajša čas, potreben za prikaz okna dnevniških zapisov in grafa revizij. Dodatna prednost je, da so ti podatki na voljo, ko stežnik ni dosegljiv.

Omogoči predpomnilnik dnevnika
: Omogoči predpomnenje dnevnika ob vsakem zahtevku. Če je možnost omogočena, se podatki - če so na voljo - preberejo iz predpomnilnika. Če podatkov ni, se preberejo iz strežnika in dodajo v predpomnilnik.

  Če je predpomnenje onemogočeno, se podatki vedno prenesejo neposredno s strežnika in se ne shranjujejo na krajevnem pogonu.

Allow ambiguous URLs
: Occasionally you may have to connect to a server which uses the same URL for all repositories. Older versions of `svnbridge` would do this. If you need to access such repositories you will have to check this option. If you don't, leave it unchecked to improve performance.

Allow ambiguous UUIDs
: Some hosting services give all their repositories the same UUID. You may even have done this yourself by copying a repository folder to create a new one. For all sorts of reasons this is a bad idea - a UUID should be *unique*. However, the log cache will still work in this situation if you check this box. If you don't need it, leave it unchecked to improve performance.

Če se s skladiščem ni mogoče povezati
: Če delate nepovezano ali če je strežnik skladišča nedostopen, lahko predpomnilnik uporabljate za pregled dnevniških zapisov, ki se že nahajajo na krajevnem pogonu. Seveda je predpomnilnik lahko zastarel, zato lahko izbirate, ali boste to možnost sploh uporabljali.

  Če so podatki o dnevniku pridobljeni brez dostopa do strežnika, je to stanje prikazano v naslovni vrstici pogovornega okna.

Časovni rok pred posodabljanjem revizije HEAD
: Ko odprete pogovorno okno dnevnika, ponavadi želite na strežniku preveriti, ali obstajajo novejši zapisi. Če je ta vrednost različna od nič, se bo preverjanje zgodilo le v primeru, da je od zadnjega preverjanja pretekel nastavljeni čas. S tem zmanjšate število dostopov do strežnika, vendar podatki morda ne bodo najsodobnejši. Če želite uporabljati to zmožnost, priporočamo uporabo vrednosti 300 (5 minut) kot dober kompromis.

Days of inactivity until small caches get removed
: If you browse around a lot of repositories you will accumulate a lot of log caches. If you're not actively using them, the cache will not grow very big, so TortoiseSVN purges them after a set time by default. Use this item to control cache purging.

Maximum size of removed inactive caches
: Larger caches are more expensive to reacquire, so TortoiseSVN only purges small caches. Fine tune the threshold with this value.

Maximum number of tool failures before cache removal
: Occasionally something goes wrong with the caching and causes a crash. If this happens the cache is normally deleted automatically to prevent a recurrence of the problem. If you use the less stable nightly build you may opt to keep the cache anyway.

4.31.7.1. Predpomnjena skladišča

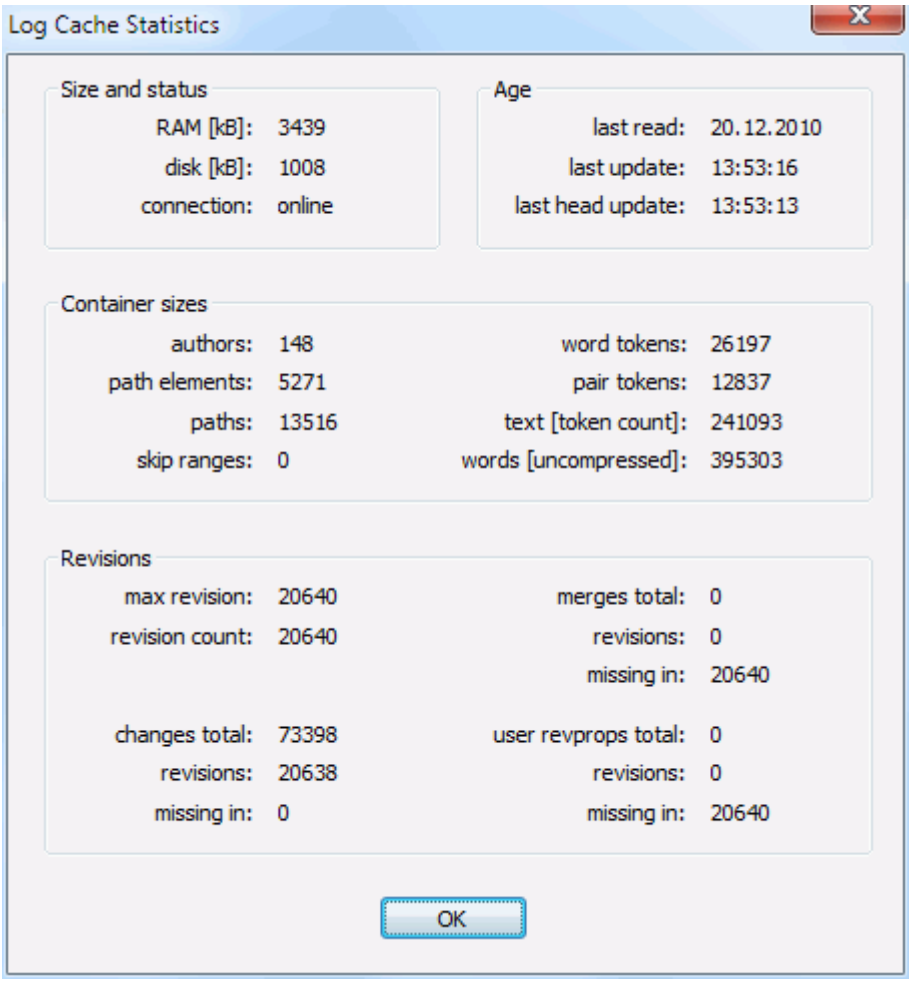
On this page you can see a list of the repositories that are cached locally, and the space used for the cache. If you select one of the repositories you can then use the buttons underneath.

Za popolno osvežitev predpomnilnika in krpanje lukenj kliknite na gumb Posodobi. Pri velikih skladiščih bo posodobitev trajala precej časa. Postopek je uporaben, če nameravate delati nepovezano in želite imeti najboljši možni predpomnilnik.

Kliknite na gumb Izvozi, če želite izvoziti celoten predpomnilnik v datoteko CSV. To je uporabno, če želite predelati podatke z uporabo zunanjega orodja. V glavnem pa ta ukaz uporabljajo razvijalci sistema.

Če želite izbrisati celoten predpomnilnik za izbrano skladišče, kliknite na gumb Izbriši. S tem ne onemogočite predpomnenja, tako da se bo ob naslednjih zahtevkih predpomnilnik ponovno pričel polniti.

4.31.7.2. Statistika predpomnilnika dnevnika



Slika 4.88. Okno nastavitev, Statistika predpomnilnika dnevnika

Za podrobno statistiko posameznega predpomnilnika kliknite na gumb Podrobnosti. Veliko polj je zanimivih zgolj za razvijalce sistema TortoiseSVN, zato le-ta niso podrobneje razložena.

RAM
: Spomin, potreben za uporabo predpomnilnika

Disk
: Velikost predpomnilnika na disku. Podatki so stisnjeni, tako da je količina porabljenega prostora dokaj skromna.

Povezava
: Pove, ali je bilo skladišče dostopno ob zadnji uporabi predpomnilnika.

Zadnja posodobitev
: Čas zadnje spremembe vsebine predpomnilnika.

Zadnja posodobitev revizije HEAD
: Čas zadnjega zahtevka revizije HEAD s strežnika

Avtorji
: Število avtorjev dnevniških zapisov v predpomnilniku.

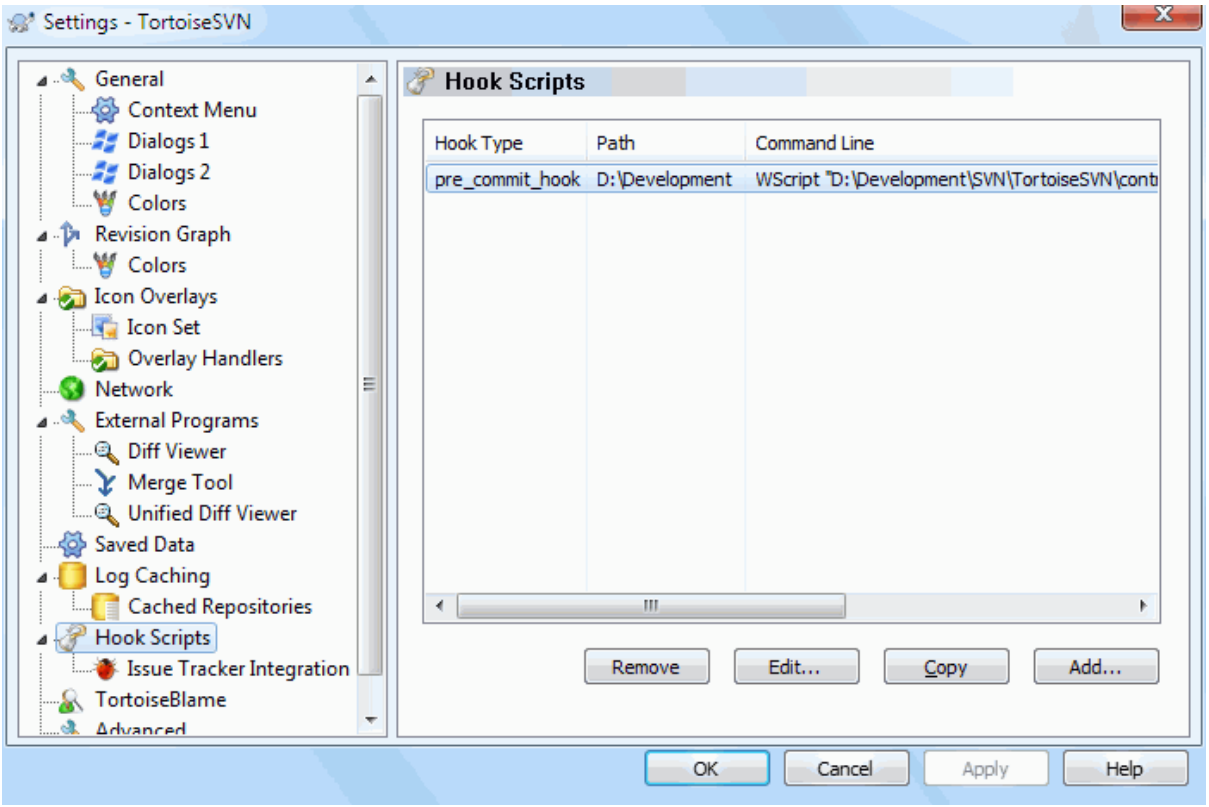
Poti
: Število poti, kot jih prikaže ukaz `svn log -v`.

Število lukenj predpomnilnika
: Število območij revizij, ki niso bila prenesena zato, ker niso bila zahtevana. To je merilo za luknjavost predpomnilnika.

Najvišja revizija
: Najvišja revizija v predpomnilniku

Število revizij
: Število revizij v predpomnilniku. To je še eno merilo popolnosti predpomnilnika.

4.31.8. Ukazne datoteke akcij na strani odjemalca

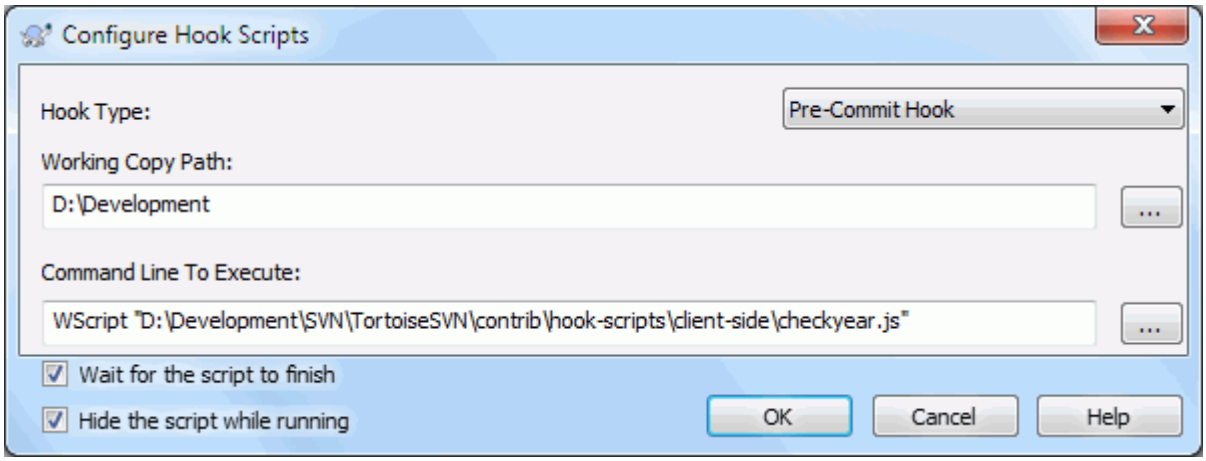


Slika 4.89. Okno nastavitev, Ukazne datoteke akcij

To okno vam omogoča nastavitev ukaznih datotek akcij, ki se izvedejo samodejno ob določeni operaciji sistema Subversion. V nasprotju s ukaznimi datotekami akcij, ki so razložene v Razdelek 3.3, “Server side hook scripts”, se te datoteke izvedejo krajevno na strani odjemalca.

Ena možnost uporabe skripte akcije je klicanje programa `SubWCRev.exe`, ki zamenja številke revizij po objavi, ali pa sproži ponovno gradnjo projekta.

Note that you can also specify such hook scripts using special properties on your working copy. See the section Razdelek 4.18.2, “Projektne lastnosti TortoiseSVN” for details.



Slika 4.90. Okno nastavitev, nastavitev ukaznih datotek akcij

Za dodajanje nove ukazne datoteke akcije preprosto kliknite na gumb Dodaj in izpolnite podrobnosti.

There are currently these types of hook script available

Akcija pred prikazom okna objave
: Called before the commit dialog is shown. You might want to use this if the hook modifies a versioned file and affects the list of files that need to be committed and/or commit message. However you should note that because the hook is called at an early stage, the full list of objects selected for commit is not available.

Manual Pre-commit
: If this is specified, the commit dialog shows a button Run Hook which when clicked runs the specified hook script. The hook script receives a list of all checked files and folders and the commit message if there was one entered.

Check-commit
: Called after the user clicks OK in the commit dialog, and before the commit dialog closes. This hook gets a list of all the checked files. If the hook returns an error, the commit dialog stays open.

  If the returned error message contains paths on newline separated lines, those paths will get selected in the commit dialog after the error message is shown.

Akcija pred objavo
: Called after the user clicks OK in the commit dialog, and before the actual commit begins. This hook has a list of exactly what will be committed.

Akcija po objavi
: Called after the commit finishes successfully.

Akcija pred prikazom okna posodobitve
: Pokliče se, preden se prikaže pogovorno okno "Posodobi na revizijo".

Akcija pred posodobitvijo
: Called before the actual Subversion update or switch begins.

Akcija po posodobitvi
: Called after the update, switch or checkout finishes (whether successful or not).

Pre-connect
: Called before an attempt to contact the repository. Called at most once in five minutes.

Pre-lock
: Called before an attempt to lock a file.

Post-lock
: Called after a file has been locked.

Ukazna datoteka akcije se definira za določeno pot v delovni kopiji. Nastaviti jo morate le za najvišjo mapo. Če operacijo izvedete na neki podmapi, bo TortoiseSVN samodejno iskal navzgor po hierarhiji.

Next you must specify the command line to execute, starting with the path to the hook script or executable. This could be a batch file, an executable file or any other file which has a valid windows file association, e.g. a perl script. Note that the script must not be specified using a UNC path as Windows shell execute will not allow such scripts to run due to security restrictions.

The command line includes several parameters which get filled in by TortoiseSVN. The parameters passed depend upon which hook is called. Each hook has its own parameters which are passed in the following order:

Akcija pred prikazom okna objave
    `PATHMESSAGEFILECWD`

Manual Pre-commit
    `PATHMESSAGEFILECWD`

Check-commit
    `PATHMESSAGEFILECWD`

Akcija pred objavo
    `PATHDEPTHMESSAGEFILECWD`

Akcija po objavi
    `PATHDEPTHMESSAGEFILEREVISIONERRORCWD`

Akcija pred prikazom okna posodobitve
    `PATHCWD`

Akcija pred posodobitvijo
    `PATHDEPTHREVISIONCWD`

Akcija po posodobitvi
    `PATHDEPTHREVISIONERRORCWDRESULTPATH`

Pre-connect
    no parameters are passed to this script. You can pass a custom parameter by appending it to the script path.

Pre-lock
    `PATHLOCKFORCEMESSAGEFILEERRORCWD`

Post-lock
    `PATHLOCKFORCEMESSAGEFILEERRORCWD`

Pomen vsakega izmed teh parametrov je podan tukaj:

PATH
    Pot do začasne datoteke, ki vsebuje vse poti, za katere je bila izvedena operacija. Vsaka pot je v svoji vrstici začasne datoteke.

    Note that for operations done remotely, e.g. in the repository browser, those paths are not local paths but the urls of the affected items.

DEPTH
    Globina, do katere se izvede objava/posodobitev.

    Možne vrednosti spremenljivk:

    -2
        `svn_depth_unknown`

    -1
        `svn_depth_exclude`

0
`svn_depth_empty`

1
`svn_depth_files`

2
`svn_depth_immediates`

3
`svn_depth_infinity`

MESSAGEFILE
Pot do datoteke, ki vsebuje dnevniški zapis za objavo. Datoteka vsebuje besedilo v obliki UTF8. Po uspešni izvršbi se besedilo dnevniškega zapisa prebere, tako da ga akcijska ukazna datoteka lahko spremeni.

REVISION
Revizija skladišča, na katero je potrebno posodobiti delovno kopijo, ali revizija po uspešni objavi.

LOCK
Either `true` when locking, or `false` when unlocking.

FORCE
Either `true` or `false`, depending on whether the operation was forced or not.

ERROR
Pot do datoteke, ki vsebuje sporočilo o napaki. Če napake ni, bo datoteka prazna.

CWD
Trenutna delovna mapa, na kateri se izvede ukazna datoteka. Nastavljena na skupno korensko mapo vseh uporabljenih poti.

RESULTPATH
A path to a temporary file which contains all the paths which were somehow touched by the operation. Each path is on a separate line in the temp file.

Note that although we have given these parameters names for convenience, you do not have to refer to those names in the hook settings. All parameters listed for a particular hook are always passed, whether you want them or not ;-)

Če želite, da operacija Subversion čaka, dokler se ukazna datoteka akcije ne zaključi, potrdite polje Počakaj, da ukazna datoteka konča izvajanje.

Običajno ne želite videti neprivlačnih oken sistema DOS, ko poženete ukazno datoteko, zato je možnost Med izvajanjem skrij ukazno datoteko po privzetih nastavitvah vključena.

Sample client hook scripts can be found in the `contrib` folder in the *TortoiseSVN repository* [https://svn.code.sf.net/p/tortoisesvn/code/trunk/contrib/hook-scripts]. (Razdelek 3, “License” explains how to access the repository.)

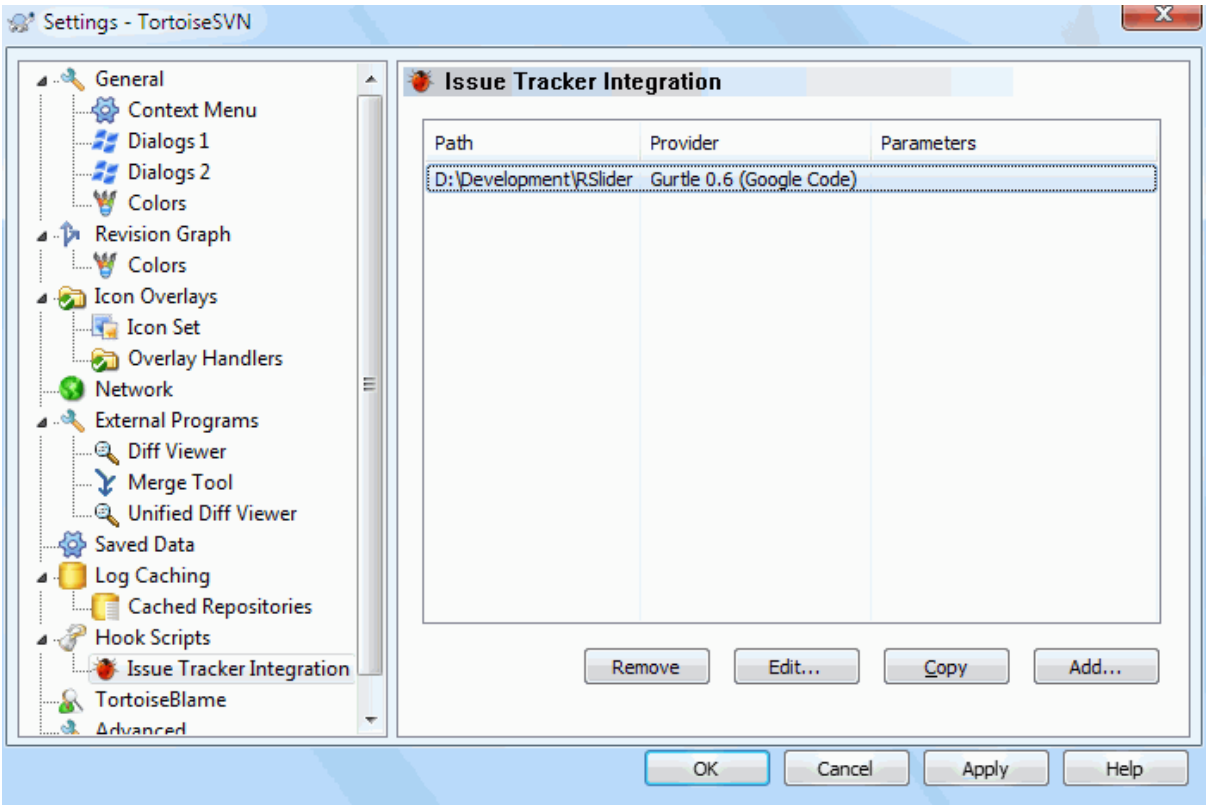
When debugging hook scripts you may want to echo progress lines to the DOS console, or insert a pause to stop the console window disappearing when the script completes. Because I/O is redirected this will not normally work. However you can redirect input and output explicitly to CON to overcome this. e.g.

```
echo Checking Status > con
pause < con > con
```

A small tool is included in the TortoiseSVN installation folder named `ConnectVPN.exe`. You can use this tool configured as a pre-connect hook to connect automatically to your VPN before TortoiseSVN tries to connect to a repository. Just pass the name of the VPN connection as the first parameter to the tool.

4.31.8.1. Integracija sledilnika zadev

TortoiseSVN lahko uporabi vtičnik COM za pregledovanje sledilnika zadev v pogovornem oknu za objave. Uporaba vtičnikov je opisana v Razdelek 4.29.2, “Pridobivanje informacij iz sledilnika zadev”. Če vam je skrbnik sistema dodelil vtičnik, ki ste ga že namestili in registrirali, sedaj tu določite, kako se vključi v delovno kopijo.



Slika 4.91. Okno za nastavitve, Okno za integracijo sledilnika zadev

Za uporabo vtičnika na določeni delovni kopiji kliknite na gumb Dodaj.... Tu določite pot delovne kopije, iz spustnega seznama registriranih vtičnikov izberete vtičnik, ki ga želite uporabiti, in navedete parametre. Parametri so odvisni od vtičnika, pametno pa je vključiti uporabniško ime sledilnika zadev, tako da vtičnik lahko poišče zadeve, ki so določene vam.

If you want all users to use the same COM plugin for your project, you can specify the plugin also with the properties `bugtraq:provideruuid`, `bugtraq:provideruuid64` and `bugtraq:providerparams`.

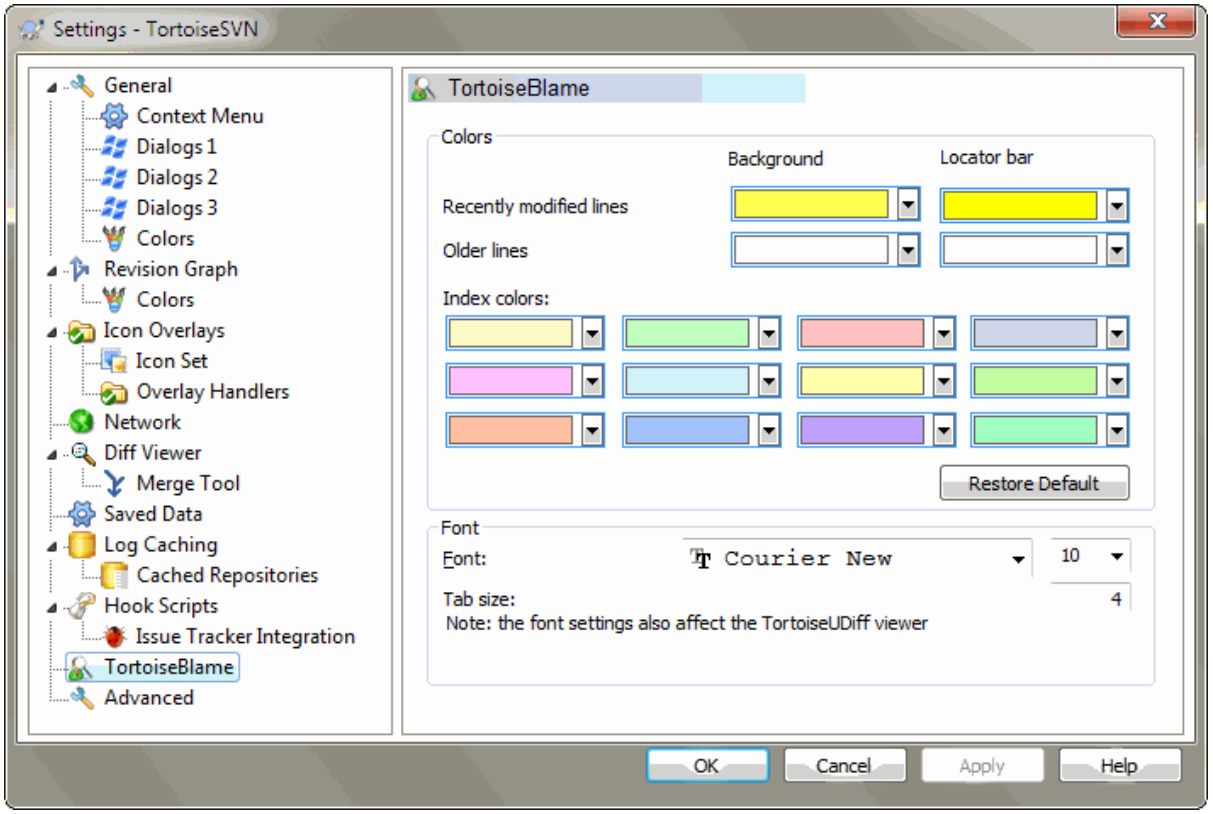
bugtraq:provideruuid
: This property specifies the COM UUID of the IBugtraqProvider, for example `{91974081-2DC7-4FB1-B3BE-0DE1C8D6CE4E}`. (This example is the UUID of the *Gurtle bugtraq provider* [http://code.google.com/p/gurtle/], which is a provider for the *Google Code* [http://code.google.com/hosting/] issue tracker.)

bugtraq:provideruuid64
: This is the same as `bugtraq:provideruuid`, but for the 64-bit version of the IBugtraqProvider.

bugtraq:providerparams
: This property specifies the parameters passed to the IBugtraqProvider.

Please check the documentation of your IBugtraqProvider plugin to find out what to specify in these two properties.

4.31.9. Nastavitve TortoiseBlame



Slika 4.92. Okno za nastavitve, TortoiseBlame

Nastavitve, ki jih uporablja TortoiseBlame, se nastavljejo v glavnem kontekstnem meniju in ne neposredno v programu TortoiseBlame.

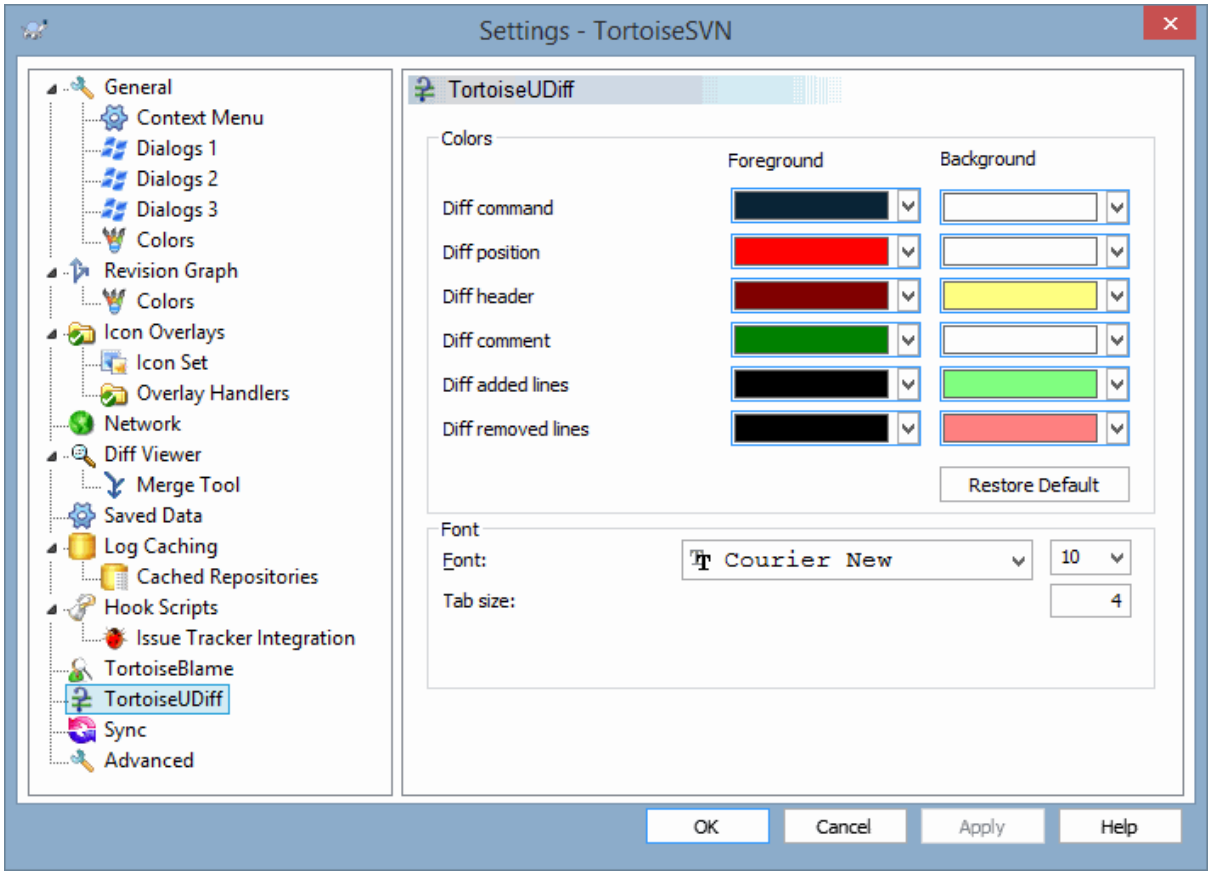
Barve
: TortoiseBlame lahko uporablja različne barve ozadja za označevanje starosti vrstic v datoteki. Nastavite začetno in končno barvo za najnovejšo in najstarejšo revizijo, TortoiseBlame pa uporabi linearno interpolacijo za določitev barve posameznih vrstic glede na številko revizije.

  You can specify different colours to use for the locator bar. The default is to use strong contrast on the locator bar while keeping the main window background light so that you can still read the text.

Pisava
: Izberete lahko tip in velikost pisave za prikazano besedilo. Nastavitev velja tako za vsebino datoteke kot za informacije na levi strani (avtor, revizija).

Tabulatorji
: Definira, koliko presledkov nadomesti tabulatorje, ki jih najde v datoteki.

4.31.10. TortoiseUDiff Settings



Slika 4.93. The Settings Dialog, TortoiseUDiff Page

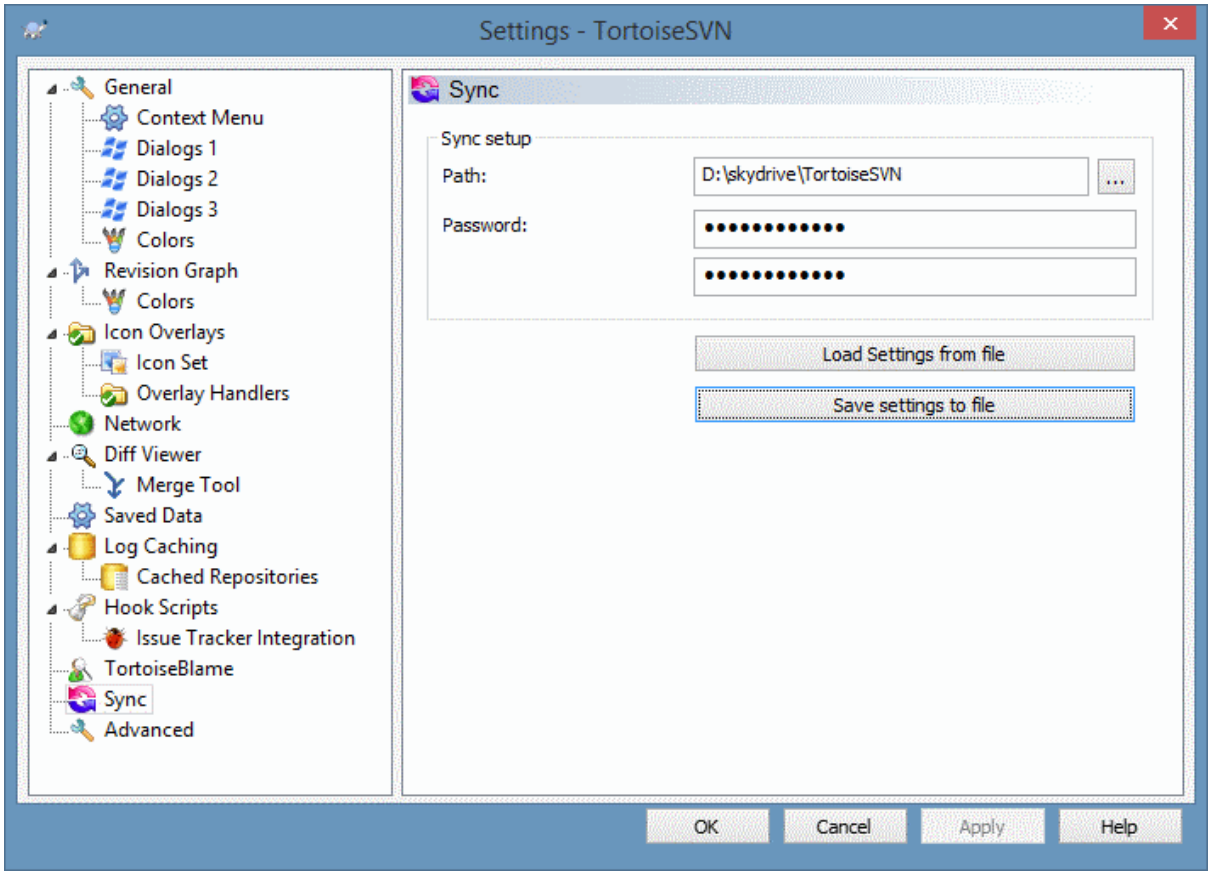
The settings used by TortoiseUDiff are controlled from the main context menu, not directly with TortoiseUDiff itself.

Barve
: The default colors used by TortoiseUDiff are usually ok, but you can configure them here.

Pisava
: You can select the font used to display the text, and the point size to use.

Tabulatorji
: Defines how many spaces to use for expansion when a tab character is found in the file diff.

4.31.11. Exporting TSVN Settings



Slika 4.94. The Settings Dialog, Sync Page

You can sync all TortoiseSVN settings to and from an encrypted file. The file is encrpyted with the password you enter so you don't have to worry if you store that file on a cloud folder like OneDrive, GDrive, DropBox, ...

When a path and password is specified, TortoiseSVN will sync all settings automatically and keep them in sync.

You can also export/import an encrypted files with all the settings manually. When you do that, you're asked for the path of the file and the password to encrypt/decrypt the settings file.

When exporting the settings manually, you can also optionally include all local settings which are not included in a normal export or in a sync. Local settings are settings which include local paths which usually vary between computers. These local settings include the configured diff and merge tools and hook scripts.

4.31.12. Advanced Settings

A few infrequently used settings are available only in the advanced page of the settings dialog. These settings modify the registry directly and you have to know what each of these settings is used for and what it does. Do not modify these settings unless you are sure you need to change them.

AllowAuthSave
: Sometimes multiple users use the same account on the same computer. In such situations it's not really wanted to save the authentication data. Setting this value to `false` disables the `save authentication` button in the authentication dialog.

AllowUnversionedObstruction
: If an update adds a new file from the repository which already exists in the local working copy as an unversioned file, the default action is to keep the local file, showing it as a (possibly) modified version of the new file from the repository. If you would prefer TortoiseSVN to create a conflict in such situations, set this value to `false`.

AlwaysExtendedMenu
: As with the explorer, TortoiseSVN shows additional commands if the **Shift** key is pressed while the context menu is opened. To force TortoiseSVN to always show those extended commands, set this value to `true`.

AutoCompleteMinChars
: The minimum amount of chars from which the editor shows an auto-completion popup. The default value is `3`.

AutocompleteRemovesExtensions
: The auto-completion list shown in the commit message editor displays the names of files listed for commit. To also include these names with extensions removed, set this value to `true`.

BlockPeggedExternals
: File externals that are pegged to a specific revision are blocked by default from being selected for a commit. This is because a subsequent update would revert those changes again unless the pegged revision of the external is adjusted.

  Set this value to `false` in case you still want to commit changes to such external files.

BlockStatus
: If you don't want the explorer to update the status overlays while another TortoiseSVN command is running (e.g. Update, Commit, ...) then set this value to `true`.

CacheTrayIcon
: To add a cache tray icon for the TSVNCache program, set this value to `true`. This is really only useful for developers as it allows you to terminate the program gracefully.

ColumnsEveryWhere
: The extra columns the TortoiseSVN adds to the details view in Windows Explorer are normally only active in a working copy. If you want those to be accessible everywhere, not just in working copies, set this value to `true`. Note that the extra columns are only available in XP. Vista and later doesn't support that feature any more. However some third-party explorer replacements do support those even on Windows versions later than XP.

ConfigDir
: You can specify a different location for the Subversion configuration file here. This will affect all TortoiseSVN operations.

CtrlEnter
: In most dialogs in TortoiseSVN, you can use **Ctrl+Enter** to dismiss the dialog as if you clicked on the OK button. If you don't want this, set this value to `false`.

Debug
: Set this to `true` if you want a dialog to pop up for every command showing the command line used to start TortoiseProc.exe.

DebugOutputString
: Set this to `true` if you want TortoiseSVN to print out debug messages during execution. The messages can be captured with special debugging tools only.

DialogTitles
: The default format (value of 0) of dialog titles is `url/path - name of dialog - TortoiseSVN`. If you set this value to 1, the format changes to `name of dialog - url/path - TortoiseSVN`.

DiffBlamesWithTortoiseMerge
: TortoiseSVN allows you to assign an external diff viewer. Most such viewers, however, are not suited for change blaming (Razdelek 4.24.2, “Okrivi spremembe”), so you might wish to fall back to TortoiseMerge in this case. To do so, set this value to `true`.

DlgStickySize
: This value specifies the number of pixels a dialog has to be near a border before the dialog sticks to it. The default value is 3. To disable this value set the value to zero.

FixCaseRenames
: Some apps change the case of filenames without notice but those changes aren't really necessary nor wanted. For example a change from `file.txt` to `FILE.TXT` wouldn't bother normal Windows applications, but Subversion is case sensitive in these situations. So TortoiseSVN automatically fixes such case changes.

  If you don't want TortoiseSVN to automatically fix such case changes for you, you can set this value to `false`.

FullRowSelect
: The status list control which is used in various dialogs (e.g., commit, check-for-modifications, add, revert, ...) uses full row selection (i.e., if you select an entry, the full row is selected, not just the first column). This is fine, but the selected row then also covers the background image on the bottom right, which can look ugly. To disable full row select, set this value to `false`.

GroupTaskbarIconsPerRepo
: This option determines how the Win7 taskbar icons of the various TortoiseSVN dialogs and windows are grouped together. This option has no effect on Vista!

1. The default value is 0. With this setting, the icons are grouped together by application type. All dialogs from TortoiseSVN are grouped together, all windows from TortoiseMerge are grouped together, ...

Slika 4.95. Taskbar with default grouping

2. If set to 1, then instead of all dialogs in one single group per application, they're grouped together by repository. For example, if you have a log dialog and a commit dialog open for repository `A`, and a check-for-modifications dialog and a log dialog for repository `B`, then there are two application icon groups shown in the Win7 taskbar, one group for each repository. But TortoiseMerge windows are not grouped together with TortoiseSVN dialogs.

Slika 4.96. Taskbar with repository grouping

3. If set to 2, then the grouping works as with the setting set to 1, except that TortoiseSVN, TortoiseMerge, TortoiseBlame, TortoiseIDiff and TortoiseUDiff windows are all grouped together. For example, if you have the commit dialog open and then double click on a modified file, the opened TortoiseMerge diff window will be put in the same icon group on the taskbar as the commit dialog icon.

Slika 4.97. Taskbar with repository grouping

4. If set to 3, then the grouping works as with the setting set to 1, but the grouping isn't done according to the repository but according to the working copy. This is useful if you have all your projects in the same repository but different working copies for each project.

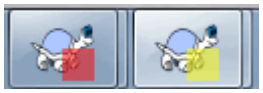
5. If set to 4, then the grouping works as with the setting set to 2, but the grouping isn't done according to the repository but according to the working copy.

HideExternalInfo
: If this is set to `false`, then every `svn:externals` is shown during an update separately.

  If it is set to `true` (the default), then update information for externals is only shown if the externals are affected by the update, i.e. changed in some way. Otherwise nothing is shown as with normal files and folders.

GroupTaskbarIconsPerRepoOverlay
: This has no effect if the option `GroupTaskbarIconsPerRepo` is set to 0 (see above).

  If this option is set to `true`, then every icon on the Win7 taskbar shows a small colored rectangle overlay, indicating the repository the dialogs/windows are used for.



Slika 4.98. Taskbar grouping with repository color overlays

IncludeExternals
: By default, TortoiseSVN always runs an update with externals included. This avoids problems with inconsistent working copies. If you have however a lot of externals set, an update can take quite a while. Set this value to `false` to run the default update with externals excluded. To update with externals included, either run the `Update to revision...` dialog or set this value to `true` again.

LogFindCopyFrom
: When the log dialog is started from the merge wizard, already merged revisions are shown in gray, but revisions beyond the point where the branch was created are also shown. These revisions are shown in black because those can't be merged.

  If this option is set to `true` then TortoiseSVN tries to find the revision where the branch was created from and hide all the revisions that are beyond that revision. Since this can take quite a while, this option is disabled by default. Also this option doesn't work with some SVN servers (e.g., Google Code Hosting, see *issue #5471* [http://code.google.com/p/support/issues/detail?id=5471]).

LogMultiRevFormat
: A format string for the log messages when multiple revisions are selected in the log dialog.

  You can use the following placeholders in your format string:

  %1!ld!
  : gets replaced with the revision number text

  %2!s!
  : gets replaced with the short log message of the revision

LogStatusCheck
: The log dialog shows the revision the working copy path is at in bold. But this requires that the log dialog fetches the status of that path. Since for very big working copies this can take a while, you can set this value to `false` to deactivate this feature.

MergeLogSeparator
: When you merge revisions from another branch, and merge tracking information is available, the log messages from the revisions you merge will be collected to make up a commit log message. A pre-defined string is

used to separate the individual log messages of the merged revisions. If you prefer, you can set this to a value containing a separator string of your choice.

NumDiffWarning
: If you want to show the diff at once for more items than specified with this settings, a warning dialog is shown first. The default is 10.

OldVersionCheck
: TortoiseSVN checks whether there's a new version available about once a week. If an updated version is found, the commit dialog shows a link control with that info. If you prefer the old behavior back where a dialog pops up notifying you about the update, set this value to `true`.

RepoBrowserTrySVNParentPath
: The repository browser tries to fetch the web page that's generated by an SVN server configured with the SVNParentPath directive to get a list of all repositories. To disable that behavior, set this value to `false`.

ScintillaDirect2D
: This option enables the use of Direct2D accelerated drawing in the Scintilla control which is used as the edit box in e.g. the commit dialog, and also for the unified diff viewer. With some graphic cards however this sometimes doesn't work properly so that the cursor to enter text isn't always visible. If that happens, you can turn this feature off by setting this value to `false`.

OutOfDateRetry
: This parameter specifies how TortoiseSVN behaves if a commit fails due to an out-of-date error:

  0
  : The user is asked whether to update the working copy or not, and the commit dialog is not reopened after the update.

  1
  : This is the default. The user is asked whether to update the working copy or not, and the commit dialog is reopened after the update so the user can proceed with the commit right away.

  2
  : Similar to `1`, but instead of updating only the paths selected for a commit, the update is done on the working copy root. This helps to avoid inconsistent working copies.

  3
  : The user is not asked to update the working copy. The commit simply fails with the out-of-date error message.

ShellMenuAccelerators
: TortoiseSVN uses accelerators for its explorer context menu entries. Since this can lead to doubled accelerators (e.g. the `SVN Commit` has the **Alt-C** accelerator, but so does the `Copy` entry of explorer). If you don't want or need the accelerators of the TortoiseSVN entries, set this value to `false`.

ShowContextMenuIcons
: This can be useful if you use something other than the windows explorer or if you get problems with the context menu displaying incorrectly. Set this value to `false` if you don't want TortoiseSVN to show icons for the shell context menu items. Set this value to `true` to show the icons again.

ShowAppContextMenuIcons
: If you don't want TortoiseSVN to show icons for the context menus in its own dialogs, set this value to `false`.

StyleCommitMessages
: The commit and log dialog use styling (e.g. bold, italic) in commit messages (see Razdelek 4.4.5, “Sporočila dnevniških zapisov objav” for details). If you don't want to do this, set the value to `false`.

UpdateCheckURL
: This value contains the URL from which TortoiseSVN tries to download a text file to find out if there are updates available. This might be useful for company admins who don't want their users to update TortoiseSVN until they approve it.

VersionCheck
: TortoiseSVN checks whether there's a new version available about once a week. If you don't want TortoiseSVN to do this check, set this value to `false`.

4.32. Zadnji korak

Darujte!

Even though TortoiseSVN and TortoiseMerge are free, you can support the developers by sending in patches and playing an active role in the development. You can also help to cheer us up during the endless hours we spend in front of our computers.

While working on TortoiseSVN we love to listen to music. And since we spend many hours on the project we need a *lot* of music. Therefore we have set up some wish-lists with our favourite music CDs and DVDs: https://tortoisesvn.net/donate.html Please also have a look at the list of people who contributed to the project by sending in patches or translations.

Poglavje 5. Project Monitor

The project monitor is a helpful tool that monitors repositories and notifies you in case there are new commits.

The projects can be monitored via a working copy path or directly via their repository URLs.

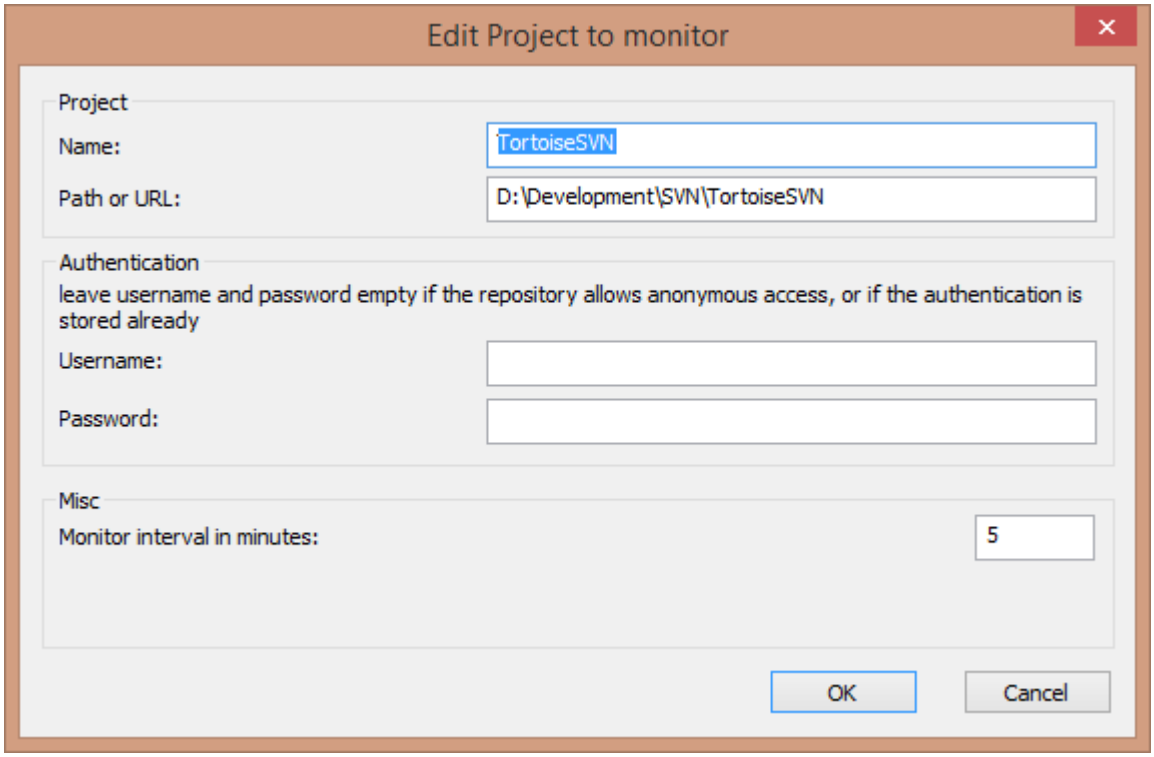
The project monitor scans each project in a configurable interval, and every time new commits are detected a notification popup is shown. Also the icon that is added to the system tray changes to indicate that there are new commits.

Snarl

If *Snarl* [https://snarl.fullphat.net/] is installed and active, then the project monitor automatically uses Snarl to show the notifications about newly detected commits.

5.1. Adding projects to monitor

If you first start the project monitor, the tree view on the left side is empty. To add projects, click on the button at the top of the dialog named Add Project.



Slika 5.1. The edit project dialog of the project monitor

To add a project for monitoring, fill in the required information. The name of the project is not optional and must be filled in, all other information is optional.

If the box for `Path or Url` is left empty, then a folder is added. This is useful to group monitored projects.

The fields `Username` and `Password` should only be filled in if the repository does not provide anonymous read access, and only if the authentication is not stored by Subversion itself. If you're accessing the monitored repository with TortoiseSVN or other svn clients and you've stored the authentication already, you should leave this empty: you won't have to edit those projects manually if the password changes.

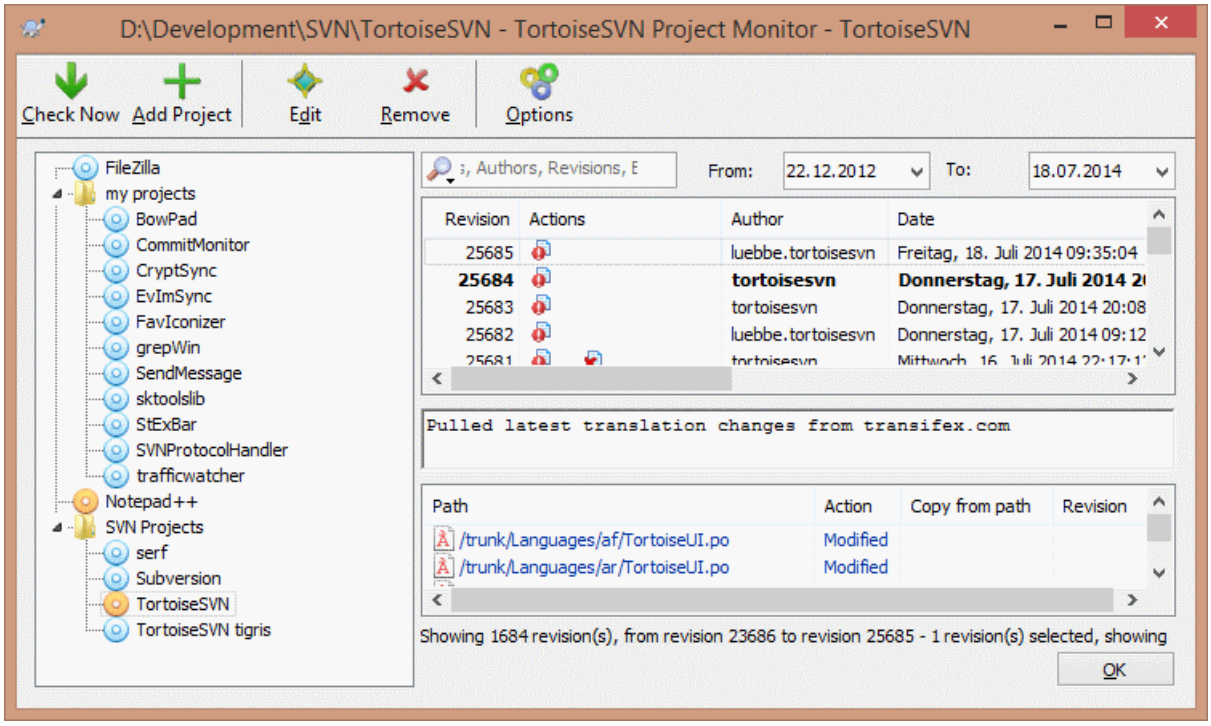
The `Monitor interval in minutes` specifies the minutes to wait in between checks. The smallest interval is one minute.

check interval

If there are a lot of users monitoring the same repository and the bandwidth on the server is limited, a repository admin can set the minimum for check intervals using an `svnrobots.txt` file. A detailed explanation on how this works can be found on the project monitor website:

http://stefanstools.sourceforge.net/svnrobots.html [http://stefanstools.sourceforge.net/svnrobots.html]

5.2. Monitor dialog



Slika 5.2. The main dialog of the project monitor

The project monitor shows all monitored projects on the left in a tree view. The projects can be moved around, for example one project can be moved below another project, making it a child/subproject.

A click on a project shows all the log messages of that project on the right.

Projects that have updates are shown in bold, with the number of new commits in brackets at the right. A click on a project marks it automatically as read.

5.2.1. Main operations

The toolbar at the top of the dialog allows to configure and operate the project monitor.

Check Now
: While each monitored project is checked according to the interval that's set up, clicking this button will force a check of all projects immediately. Note that if there are updates, the notification won't show up until all projects have been checked.

Add Project
: Opens a new dialog to set up a new project for monitoring.

Uredi
: Opens the configuration dialog for the selected project.

Odstrani
: Removes the selected project after a confirmation dialog is shown.

Mark all as read
: Marks all revisions in all projects as read. Note that if you select a project with unread revisions, those revisions are automatically marked as read when you select another project.

  If you hold down the **Shift** key when clicking the button, all error states are also cleared if there are any.

Update all
: Runs an **Update** on all monitored working copies. Projects that are monitored via an url are not updated, only those that are set up with a working copy path.

Možnosti
: Shows a dialog to configure the behavior of the project monitor.

Poglavje 6. Program SubWCRev

SubWCRev je konzolna aplikacija, ki se uporablja za pridobivanje informacij o stanju delovne kopije z možnostjo zamenjave ključnih besed v predlogi. Postopek se pogosto uporablja kot del procesa gradnje projekta. S tem se v elemente projekta, ki se gradi, vnese informacije o delovni kopiji. Tipičen primer uporabe je prikaz številke revizije v pogovornem oknu “O programu”.

6.1. Program SubWCRev za ukazno vrtico

SubWCRev reads the Subversion status of all files in a working copy, excluding externals by default. It records the highest commit revision number found, and the commit timestamp of that revision, it also records whether there are local modifications in the working copy, or mixed update revisions. The revision number, update revision range and modification status are displayed on stdout.

SubWCRev.exe is called from the command line or a script, and is controlled using the command line parameters.

```
SubWCRev WorkingCopyPath [SrcVersionFile DstVersionFile] [-nmdfe]
```

`PotDelovneKopije` je pot do delovne kopije, ki se preverja. SubWCRev lahko uporabljate le na delovni kopiji, ne pa tudi neposredno na skladišču. Pot je lahko absolutna ali relativna na trenutno mapo.

Če želite, da SubWCRev opravi zamenjavo ključnih besed, tako da so polja revizija in naslov URL shranjena v besedilni datoteki, morate podati predlogo `IzvirnaDatoteka` in izhodno datoteko `CiljnaDatoteka`, ki vsebuje zamenjano verzijo predloge.

You can specify ignore patterns for SubWCRev to prevent specific files and paths from being considered. The patterns are read from a file named `.subwcrevignore`. The file is read from the specified path, and also from the working copy root. If the file does not exist, no files or paths are ignored. The `.subwcrevignore` file can contain multiple patterns, separated by newlines. The patterns are matched against the paths relative to the repository root and paths relative to the path of the `.subwcrevignore` file. For example, to ignore all files in the `doc` folder of the TortoiseSVN working copy, the `.subwcrevignore` would contain the following lines:

```
/trunk/doc
/trunk/doc/*
```

Or, assuming the `.subwcrevignore` file is in the working copy root which is checked out from trunk, using the patterns

```
doc
doc/*
```

is the same as the example above.

To ignore all images, the ignore patterns could be set like this:

```
*.png
*.jpg
*.ico
*.bmp
```

Pomembno

The ignore patterns are case-sensitive, just like Subversion is.

Namig

To create a file with a starting dot in the Windows explorer, enter `.subwcrevignore.`. Note the trailing dot.

There are a number of optional switches which affect the way SubWCRev works. If you use more than one, they must be specified as a single group, e.g. `-nm`, not `-n -m`.

Preklop	Opis
-n	Ob uporabi tega stikala se SubWCRev zaključi s številko napake `ERRORLEVEL 7`, če delovna kopija vsebuje krajevne spremembe. To stikalo se uporablja, da se prepreči gradnja projekta, ki vsebuje neobjavljene spremembe.
-N	If this switch is given, SubWCRev will exit with `ERRORLEVEL 11` if the working copy contains unversioned items that are not ignored.
-m	Ob uporabi tega stikala se SubWCRev zaključi s številko napake `ERRORLEVEL 8`, če delovna kopija vsebuje mešane revizije. To stikalo se uporablja, da se prepreči gradnja projekta, ki vsebuje delno posodobljeno delovno kopijo.
-d	Ob uporabi tega stikala se SubWCRev zaključi s številko napake `ERRORLEVEL 9`, če ciljna datoteka že obstaja.
-f	Ob uporabi tega stikala SubWCRev pri iskanju najvišje revizije vključi tudi mape. Po privzetih nastavitvah se išče najvišja revizija le na datotekah.
-e	Ob uporabi tega stikala SubWCRev pregleda mape, ki so označene z lastnostjo `svn:externals`, vendar le, če se nahajajo v istem skladišču. Privzeta nastavitev ne upošteva zunanjih datotek.
-E	If this switch is given, same as `-e`, but it ignores the externals with explicit revisions, when the revision range inside of them is only the given explicit revision in the properties. So it doesn't lead to mixed revisions.
-x	Ob uporabi tega stikala SubWCRev izpiše številke revizij v šestnajstiški obliki
-X	Ob uporabi tega stikala SubWCRev izpiše številke revizij v šestnajstiški obliki s predpono '0X'.
-F	If this switch is given, SubWCRev will ignore any `.subwcrevignore` files and include all files.
-q	If this switch is given, SubWCRev will perform the keyword substitution without showing working copy status on stdout.

Tabela 6.1. Seznam stikal ukazne vrstice, ki so na voljo

If there is no error, SubWCRev returns zero. But in case an error occurs, the error message is written to stderr and shown in the console. And the returned error codes are:

Error Code	Opis
1	Syntax error. One or more command line parameters are invalid.
2	The file or folder specified on the command line was not found.
3	The input file could not be opened, or the target file could not be created.
4	Could not allocate memory. This could happen if e.g. the source file is too big.
5	The source file can not be scanned properly.
6	SVN error: Subversion returned with an error when SubWCRev tried to find the information from the working copy.
7	The working copy has local modifications. This requires the `-n` switch.

Error Code	Opis
8	The working copy has mixed revisions. This requires the `-m` switch.
9	The output file already exists. This requires the `-d` switch.
10	The specified path is not a working copy or part or one.
11	The working copy has unversioned files or folders in it. This requires the `-N` switch.

Tabela 6.2. List of SubWCRev error codes

6.2. Zamenjava ključnih besed

Če podamo začetno in ciljno datoteko, SubWCRev prekopira začetno datoteko v ciljno, pri tem pa izvede zamenjavo ključnih besed:

Ključna beseda	Opis
$WCREV$	Zamenjano z najvišjo revizijo v delovni kopiji.
$WCREV&$	Replaced with the highest commit revision in the working copy, ANDed with the value after the & char. For example: `$WCREV&0xFFFF$`
$WCREV-$, $WCREV+$	Replaced with the highest commit revision in the working copy, with the value after the + or - char added or subtracted. For example: `$WCREV-1000$`
$WCDATE$, $WCDATEUTC$	Replaced with the commit date/time of the highest commit revision. By default, international format is used: `yyyy-mm-dd hh:mm:ss`. Alternatively, you can specify a custom format which will be used with `strftime()`, for example: `$WCDATE=%a %b %d %I:%M:%S %p$`. For a list of available formatting characters, look at the *spletna navodila* [http://msdn.microsoft.com/en-us/library/fe06s4ak.aspx].
$WCNOW$, $WCNOWUTC$	Replaced with the current system date/time. This can be used to indicate the build time. Time formatting can be used as described for `$WCDATE$`.
$WCRANGE$	Replaced with the update revision range in the working copy. If the working copy is in a consistent state, this will be a single revision. If the working copy contains mixed revisions, either due to being out of date, or due to a deliberate update-to-revision, then the range will be shown in the form 100:200.
$WCMIXED$	`$WCMIXED?TText:FText$` se zamenja z vrednostjo `TText`, če obstajajo mešane revizije, in z vrednostjo `FText`, če ne obstajajo.
$WCMODS$	`$WCMODS?TText:FText$` se zamenja z vrednostjo `TText`, če obstajajo krajevne spremembe, in z vrednostjo `FText`, če krajevnih sprememb ni.
$WCUNVER$	`$WCUNVER?TText:FText$` is replaced with `TText` if there are unversioned items in the working copy, or `FText` if not.
$WCEXTALLFIXED$	`$WCEXTALLFIXED?TText:FText$` is replaced with `TText` if all externals are fixed to an explicit revision, or `FText` if not.
$WCISTAGGED$	`$WCISTAGGED?TText:FText$` is replaced with `TText` if the repository URL contains the tags classification pattern, or `FText` if not.
$WCURL$	Zamenjano z naslovom URL delovne kopije, ki je bil podan programu SubWCRev.
$WCINSVN$	`$WCINSVN?TText:FText$` se zamenja z vrednostjo `TText`, če je element pod nadzorom, in z vrednostjo `FText`, če ni.
$WCNEEDSLOCK$	`$WCNEEDSLOCK?TText:FText$` se zamenja z vrednostjo `TText`, če ima element nastavljeno lastnost `svn:needs-lock`, in z vrednostjo `FText`, če je nima.
$WCISLOCKED$	`$WCISLOCKED?TText:FText$` se zamenja z vrednostjo `TText`, če je element zaklenjen, in z vrednostjo `FText`, če ni.

Ključna beseda	Opis
$WCLOCKDATE$, $WCLOCKDATEUTC$	Replaced with the lock date. Time formatting can be used as described for `$WCDATE$`.
$WCLOCKOWNER$	Replaced with the name of the lock owner.
$WCLOCKCOMMENT$	Replaced with the comment of the lock.
$WCUNVER$	`$WCUNVER?TText:FText$` is replaced with `TText` if there are unversioned files or folders in the working copy, or `FText` if not.

Tabela 6.3. List of available keywords

SubWCRev does not directly support nesting of expressions, so for example you cannot use an expression like:

```
#define SVN_REVISION    "$WCMIXED?$WCRANGE$:$WCREV$$"
```

But you can usually work around it by other means, for example:

```
#define SVN_RANGE       $WCRANGE$
#define SVN_REV         $WCREV$
#define SVN_REVISION    "$WCMIXED?SVN_RANGE:SVN_REV$"
```

Namig

Some of these keywords apply to single files rather than to an entire working copy, so it only makes sense to use these when SubWCRev is called to scan a single file. This applies to `$WCINSVN$`, `$WCNEEDSLOCK$`, `$WCISLOCKED$`, `$WCLOCKDATE$`, `$WCLOCKOWNER$` and `$WCLOCKCOMMENT$`.

6.3. Primer uprabe ključne besede

Spodnji primer prikazuje, kako se ključne besede v predlogi zamenjajo v izhodni datoteki.

```
// Test file for SubWCRev

char *Revision      = "$WCREV$";
char *Revision16    = "$WCREV&0xFF$";
char *Revisionp100  = "$WCREV+100$";
char *Revisionm100  = "$WCREV-100$";
char *Modified      = "$WCMODS?Modified:Not modified$";
char *Unversioned   = "$WCUNVER?Unversioned items found:no unversioned items$";
char *Date          = "$WCDATE$";
char *CustDate      = "$WCDATE=%a, %d %B %Y$";
char *DateUTC       = "$WCDATEUTC$";
char *CustDateUTC   = "$WCDATEUTC=%a, %d %B %Y$";
char *TimeNow       = "$WCNOW$";
char *TimeNowUTC    = "$WCNOWUTC$";
char *RevRange      = "$WCRANGE$";
char *Mixed         = "$WCMIXED?Mixed revision WC:Not mixed$";
char *ExtAllFixed   = "$WCEXTALLFIXED?All externals fixed:Not all externals fixed$";
char *IsTagged      = "$WCISTAGGED?Tagged:Not tagged$";
char *URL           = "$WCURL$";
char *isInSVN       = "$WCINSVN?versioned:not versioned$";
```

```
char *needslck      = "$WCNEEDSLOCK?TRUE:FALSE$";
char *islocked      = "$WCISLOCKED?locked:not locked$";
char *lockdateutc   = "$WCLOCKDATEUTC$";
char *lockdate      = "$WCLOCKDATE$";
char *lockcustutc   = "$WCLOCKDATEUTC=%a, %d %B %Y$";
char *lockcust      = "$WCLOCKDATE=%a, %d %B %Y$";
char *lockown       = "$WCLOCKOWNER$";
char *lockcmt       = "$WCLOCKCOMMENT$";

#if $WCMODS?1:0$
#error Source is modified
#endif

// End of file
```

Po zagonu programa `SubWCRev.exe pot\do\delovnekopije testfile.tmpl testfile.txt` bo izhodna datoteka `testfile.txt` izgledala takole:

```
// Test file for SubWCRev

char *Revision      = "22837";
char *Revision16    = "53";
char *Revisionp100  = "22937";
char *Revisionm100  = "22737";
char *Modified      = "Modified";
char *Unversioned   = "no unversioned items";
char *Date          = "2012/04/26 18:47:57";
char *CustDate      = "Thu, 26 April 2012";
char *DateUTC       = "2012/04/26 16:47:57";
char *CustDateUTC   = "Thu, 26 April 2012";
char *TimeNow       = "2012/04/26 20:51:17";
char *TimeNowUTC    = "2012/04/26 18:51:17";
char *RevRange      = "22836:22837";
char *Mixed         = "Mixed revision WC";
char *ExtAllFixed   = "All externals fixed";
char *IsTagged      = "Not tagged";
char *URL           = "https://svn.code.sf.net/p/tortoisesvn/code/trunk";
char *isInSVN       = "versioned";
char *needslck      = "FALSE";
char *islocked      = "not locked";
char *lockdateutc   = "1970/01/01 00:00:00";
char *lockdate      = "1970/01/01 01:00:00";
char *lockcustutc   = "Thu, 01 January 1970";
char *lockcust      = "Thu, 01 January 1970";
char *lockown       = "";
char *lockcmt       = "";

#if 1
#error Source is modified
#endif

// End of file
```

Namig

Takšna datoteka je vključena v proces gradnje, zato bi pričakovali, da bo pod nadzorom. Pod nadzor dajte predlogo datoteke in ne samodejno ustvarjene datoteke, v nasprotnem primeru boste morali

ob vsakem usvarjanju datoteke narediti objavo, kar pa bi pomenilo, da je potrebno datoteko spet posodobiti.

6.4. Vmesnik COM

Če morate do informacij o reviziji dostopati iz drugih programov, lahko uporabite vmesnik COM programa SubWCRev. Ustvariti morate objekt `SubWCRev.object`, na katerem so podprte naslednje metode:

Način	Opis
.GetWCInfo	Metoda pregleda delovno kopijo in pridobi informacije o revizijah. Klicati jo morate pred metodami, ki so opisane v nadaljevanju. Prvi parameter je pot. Drugi parameter mora imeti vrednost true, če želite upoštevati tudi revizije map. Ta parameter je ekvivalenten stikalu `-f` v ukazni vrstici. Tretji parameter mora biti true, če želite vključiti tudi zunanje elemente. Ta paramter je ekvivalenten stikalu `-e` v ukazni vrstici.
.GetWCInfo2	The same as `GetWCInfo()` but with a fourth parameter that sets the equivalent to the `-E` command line switch.
.Revision	The highest commit revision in the working copy. Equivalent to `$WCREV$`.
.Date	The commit date/time of the highest commit revision. Equivalent to `$WCDATE$`.
.Author	Avtor najvišje objavljene revizije, to je zadnji avtor, ki je objavil spremembe delovne kopije.
.MinRev	Najnižja revizija posodobitve, kot je prikazana z `$WCRANGE$`
.MaxRev	Najvišja revizija posodobitve, kot je prikazana z `$WCRANGE$`
.HasModifications	'True', če obstajajo krajevne spremembe
.HasUnversioned	True if there are unversioned items
.Url	Replaced with the repository URL of the working copy path used in `GetWCInfo`. Equivalent to `$WCURL$`.
.IsSvnItem	True if the item is versioned.
.NeedsLocking	True if the item has the `svn:needs-lock` property set.
.IsLocked	True if the item is locked.
.LockCreationDate	String representing the date when the lock was created, or an empty string if the item is not locked.
.LockOwner	String representing the lock owner, or an empty string if the item is not locked.
.LockComment	The message entered when the lock was created.

Tabela 6.4. Podprte COM/avtomatizacijske metode

Naslednji primer prikazuje uporabo vmesnika.

```
// testCOM.js - javascript file
// test script for the SubWCRev COM/Automation-object

filesystem = new ActiveXObject("Scripting.FileSystemObject");

revObject1 = new ActiveXObject("SubWCRev.object");
revObject2 = new ActiveXObject("SubWCRev.object");
revObject3 = new ActiveXObject("SubWCRev.object");
revObject4 = new ActiveXObject("SubWCRev.object");

revObject1.GetWCInfo(
```

```
    filesystem.GetAbsolutePathName("."), 1, 1);
revObject2.GetWCInfo(
    filesystem.GetAbsolutePathName(".."), 1, 1);
revObject3.GetWCInfo(
    filesystem.GetAbsolutePathName("SubWCRev.cpp"), 1, 1);
revObject4.GetWCInfo2(
    filesystem.GetAbsolutePathName("..\\.."), 1, 1, 1);

wcInfoString1 = "Revision = " + revObject1.Revision +
                "\nMin Revision = " + revObject1.MinRev +
                "\nMax Revision = " + revObject1.MaxRev +
                "\nDate = " + revObject1.Date +
                "\nURL = " + revObject1.Url + "\nAuthor = " +
                revObject1.Author + "\nHasMods = " +
                revObject1.HasModifications + "\nIsSvnItem = " +
                revObject1.IsSvnItem + "\nNeedsLocking = " +
                revObject1.NeedsLocking + "\nIsLocked = " +
                revObject1.IsLocked + "\nLockCreationDate = " +
                revObject1.LockCreationDate + "\nLockOwner = " +
                revObject1.LockOwner + "\nLockComment = " +
                revObject1.LockComment;
wcInfoString2 = "Revision = " + revObject2.Revision +
                "\nMin Revision = " + revObject2.MinRev +
                "\nMax Revision = " + revObject2.MaxRev +
                "\nDate = " + revObject2.Date +
                "\nURL = " + revObject2.Url + "\nAuthor = " +
                revObject2.Author + "\nHasMods = " +
                revObject2.HasModifications + "\nIsSvnItem = " +
                revObject2.IsSvnItem + "\nNeedsLocking = " +
                revObject2.NeedsLocking + "\nIsLocked = " +
                revObject2.IsLocked + "\nLockCreationDate = " +
                revObject2.LockCreationDate + "\nLockOwner = " +
                revObject2.LockOwner + "\nLockComment = " +
                revObject2.LockComment;
wcInfoString3 = "Revision = " + revObject3.Revision +
                "\nMin Revision = " + revObject3.MinRev +
                "\nMax Revision = " + revObject3.MaxRev +
                "\nDate = " + revObject3.Date +
                "\nURL = " + revObject3.Url + "\nAuthor = " +
                revObject3.Author + "\nHasMods = " +
                revObject3.HasModifications + "\nIsSvnItem = " +
                revObject3.IsSvnItem + "\nNeedsLocking = " +
                revObject3.NeedsLocking + "\nIsLocked = " +
                revObject3.IsLocked + "\nLockCreationDate = " +
                revObject3.LockCreationDate + "\nLockOwner = " +
                revObject3.LockOwner + "\nLockComment = " +
                revObject3.LockComment;
wcInfoString4 = "Revision = " + revObject4.Revision +
                "\nMin Revision = " + revObject4.MinRev +
                "\nMax Revision = " + revObject4.MaxRev +
                "\nDate = " + revObject4.Date +
                "\nURL = " + revObject4.Url + "\nAuthor = " +
                revObject4.Author + "\nHasMods = " +
                revObject4.HasModifications + "\nIsSvnItem = " +
                revObject4.IsSvnItem + "\nNeedsLocking = " +
                revObject4.NeedsLocking + "\nIsLocked = " +
                revObject4.IsLocked + "\nLockCreationDate = " +
                revObject4.LockCreationDate + "\nLockOwner = " +
```

```
                revObject4.LockOwner + "\nLockComment = " +
                revObject4.LockComment;

WScript.Echo(wcInfoString1);
WScript.Echo(wcInfoString2);
WScript.Echo(wcInfoString3);
WScript.Echo(wcInfoString4);
```

The following listing is an example on how to use the SubWCRev COM object from C#:

```
using LibSubWCRev;
SubWCRev sub = new SubWCRev();
sub.GetWCInfo("C:\\PathToMyFile\\MyFile.cc", true, true);
if (sub.IsSvnItem == true)
{
    MessageBox.Show("versioned");
}
else
{
    MessageBox.Show("not versioned");
}
```

Poglavje 7. IBugtraqProvider interface

To get a tighter integration with issue trackers than by simply using the `bugtraq:` properties, TortoiseSVN can make use of COM plugins. With such plugins it is possible to fetch information directly from the issue tracker, interact with the user and provide information back to TortoiseSVN about open issues, verify log messages entered by the user and even run actions after a successful commit to e.g, close an issue.

We can't provide information and tutorials on how you have to implement a COM object in your preferred programming language, but we have example plugins in C++/ATL and C# in our repository in the `contrib/issue-tracker-plugins` folder. In that folder you can also find the required include files you need to build your plugin. (Razdelek 3, “License” explains how to access the repository.)

Pomembno

You should provide both a 32-bit and 64-bit version of your plugin. Because the x64-Version of TortoiseSVN can not use a 32-bit plugin and vice-versa.

7.1. Naming conventions

If you release an issue tracker plugin for TortoiseSVN, please do *not* name it *Tortoise<Something>*. We'd like to reserve the *Tortoise* prefix for a version control client integrated into the windows shell. For example: TortoiseCVS, TortoiseSVN, TortoiseHg, TortoiseGit and TortoiseBzr are all version control clients.

Please name your plugin for a Tortoise client *Turtle<Something>*, where *<Something>* refers to the issue tracker that you are connecting to. Alternatively choose a name that sounds like *Turtle* but has a different first letter. Nice examples are:

- Gurtle - An issue tracker plugin for Google code
- TurtleMine - An issue tracker plugin for Redmine
- VurtleOne - An issue tracker plugin for VersionOne

7.2. The IBugtraqProvider interface

TortoiseSVN 1.5 and later can use plugins which implement the IBugtraqProvider interface. The interface provides a few methods which plugins can use to interact with the issue tracker.

```
HRESULT ValidateParameters (
  // Parent window for any UI that needs to be
  // displayed during validation.
  [in] HWND hParentWnd,

  // The parameter string that needs to be validated.
  [in] BSTR parameters,

  // Is the string valid?
  [out, retval] VARIANT_BOOL *valid
);
```

This method is called from the settings dialog where the user can add and configure the plugin. The `parameters` string can be used by a plugin to get additional required information, e.g., the URL to the issue tracker, login information, etc. The plugin should verify the `parameters` string and show an error dialog if the string is not valid. The `hParentWnd` parameter should be used for any dialog the plugin shows as the parent window. The plugin must return TRUE if the validation of the `parameters` string is successful. If the plugin returns FALSE, the settings dialog won't allow the user to add the plugin to a working copy path.

```
HRESULT GetLinkText (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The parameter string, just in case you need to talk to your
  // web service (e.g.) to find out what the correct text is.
  [in] BSTR parameters,

  // What text do you want to display?
  // Use the current thread locale.
  [out, retval] BSTR *linkText
);
```

The plugin can provide a string here which is used in the TortoiseSVN commit dialog for the button which invokes the plugin, e.g., "Choose issue" or "Select ticket". Make sure the string is not too long, otherwise it might not fit into the button. If the method returns an error (e.g., `E_NOTIMPL`), a default text is used for the button.

```
HRESULT GetCommitMessage (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // The new text for the commit message.
  // This replaces the original message.
  [out, retval] BSTR *newMessage
);
```

This is the main method of the plugin. This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button.

The `parameters` string is the string the user has to enter in the settings dialog when he configures the plugin. Usually a plugin would use this to find the URL of the issue tracker and/or login information or more.

The `commonRoot` string contains the parent path of all items selected to bring up the commit dialog. Note that this is *not* the root path of all items which the user has selected in the commit dialog. For the branch/tag dialog, this is the path which is to be copied.

The `pathList` parameter contains an array of paths (as strings) which the user has selected for the commit.

The `originalMessage` parameter contains the text entered in the log message box in the commit dialog. If the user has not yet entered any text, this string will be empty.

The `newMessage` return string is copied into the log message edit box in the commit dialog, replacing whatever is already there. If a plugin does not modify the `originalMessage` string, it must return the same string again here, otherwise any text the user has entered will be lost.

7.3. The IBugtraqProvider2 interface

In TortoiseSVN 1.6 a new interface was added which provides more functionality for plugins. This IBugtraqProvider2 interface inherits from IBugtraqProvider.

```
HRESULT GetCommitMessage2 (
  // Parent window for your provider's UI.
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,
  // The common URL of the commit
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,

  // The text already present in the commit message.
  // Your provider should include this text in the new message,
  // where appropriate.
  [in] BSTR originalMessage,

  // You can assign custom revision properties to a commit
  // by setting the next two params.
  // note: Both safearrays must be of the same length.
  //       For every property name there must be a property value!

  // The content of the bugID field (if shown)
  [in] BSTR bugID,

  // Modified content of the bugID field
  [out] BSTR * bugIDOut,

  // The list of revision property names.
  [out] SAFEARRAY(BSTR) * revPropNames,

  // The list of revision property values.
  [out] SAFEARRAY(BSTR) * revPropValues,

  // The new text for the commit message.
  // This replaces the original message
  [out, retval] BSTR * newMessage
);
```

This method is called from the TortoiseSVN commit dialog when the user clicks on the plugin button. This method is called instead of `GetCommitMessage()`. Please refer to the documentation for `GetCommitMessage` for the parameters that are also used there.

The parameter `commonURL` is the parent URL of all items selected to bring up the commit dialog. This is basically the URL of the `commonRoot` path.

The parameter `bugID` contains the content of the bug-ID field (if it is shown, configured with the property `bugtraq:message`).

The return parameter `bugIDOut` is used to fill the bug-ID field when the method returns.

The `revPropNames` and `revPropValues` return parameters can contain name/value pairs for revision properties that the commit should set. A plugin must make sure that both arrays have the same size on return! Each property name in `revPropNames` must also have a corresponding value in `revPropValues`. If no revision properties are to be set, the plugin must return empty arrays.

```
HRESULT CheckCommit (
  [in] HWND hParentWnd,
```

```
  [in] BSTR parameters,
  [in] BSTR commonURL,
  [in] BSTR commonRoot,
  [in] SAFEARRAY(BSTR) pathList,
  [in] BSTR commitMessage,
  [out, retval] BSTR * errorMessage
);
```

This method is called right before the commit dialog is closed and the commit begins. A plugin can use this method to validate the selected files/folders for the commit and/or the commit message entered by the user. The parameters are the same as for `GetCommitMessage2()`, with the difference that `commonURL` is now the common URL of all *checked* items, and `commonRoot` the root path of all checked items.

For the branch/tag dialog, the `commonURL` is the source URL of the copy, and `commonRoot` is set to the target URL of the copy.

The return parameter `errorMessage` must either contain an error message which TortoiseSVN shows to the user or be empty for the commit to start. If an error message is returned, TortoiseSVN shows the error string in a dialog and keeps the commit dialog open so the user can correct whatever is wrong. A plugin should therefore return an error string which informs the user *what* is wrong and how to correct it.

```
HRESULT  OnCommitFinished (
  // Parent window for any (error) UI that needs to be displayed.
  [in] HWND hParentWnd,

  // The common root of all paths that got committed.
  [in] BSTR commonRoot,

  // All the paths that got committed.
  [in] SAFEARRAY(BSTR) pathList,


  // The text already present in the commit message.
  [in] BSTR logMessage,

  // The revision of the commit.
  [in] ULONG revision,


  // An error to show to the user if this function
  // returns something else than S_OK
  [out, retval] BSTR * error
);
```

This method is called after a successful commit. A plugin can use this method to e.g., close the selected issue or add information about the commit to the issue. The parameters are the same as for `GetCommitMessage2`.

```
HRESULT HasOptions(
  // Whether the provider provides options
  [out, retval] VARIANT_BOOL *ret
);
```

This method is called from the settings dialog where the user can configure the plugins. If a plugin provides its own configuration dialog with `ShowOptionsDialog`, it must return TRUE here, otherwise it must return FALSE.

```
HRESULT ShowOptionsDialog(
```

```
  // Parent window for the options dialog
  [in] HWND hParentWnd,

  // Parameters for your provider.
  [in] BSTR parameters,

  // The parameters string
  [out, retval] BSTR * newparameters
);
```

This method is called from the settings dialog when the user clicks on the "Options" button that is shown if `HasOptions` returns TRUE. A plugin can show an options dialog to make it easier for the user to configure the plugin.

The `parameters` string contains the plugin parameters string that is already set/entered.

The `newparameters` return parameter must contain the parameters string which the plugin constructed from the info it gathered in its options dialog. That `paramameters` string is passed to all other IBugtraqProvider and IBugtraqProvider2 methods.

Dodatek A. Pogosto zastavljena vprašanja (FAQ)

Because TortoiseSVN is being developed all the time it is sometimes hard to keep the documentation completely up to date. We maintain an *online FAQ* [https://tortoisesvn.net/faq.html] which contains a selection of the questions we are asked the most on the TortoiseSVN mailing lists <dev@tortoisesvn.tigris.org> and <users@tortoisesvn.tigris.org>.

We also maintain a project *Issue Tracker* [https://sourceforge.net/p/tortoisesvn/tickets/] which tells you about some of the things we have on our To-Do list, and bugs which have already been fixed. If you think you have found a bug, or want to request a new feature, check here first to see if someone else got there before you.

If you have a question which is not answered anywhere else, the best place to ask it is on one of the mailing lists:

- <users@tortoisesvn.tigris.org> is the one to use if you have questions about using TortoiseSVN.
- If you want to help out with the development of TortoiseSVN, then you should take part in discussions on <dev@tortoisesvn.tigris.org>.
- If you want to help with the translation of the TortoiseSVN user interface or the documentation, send an e-mail to <translators@tortoisesvn.tigris.org>.

Dodatek B. Kako naredim...

Ta dodatek vsebuje rešitve težav/vprašanj, ki se lahko pojavijo pri uporabi TortoiseSVN.

B.1. Kako premaknem/kopiram večje število datotek naenkrat

Premikanje/kopiranje posameznih datotek se izvaja z ukazom TortoiseSVN → Preimenuj.... Če premikate/kopirate večje število datotek, je ta postopek počasen in zahteva preveč dela.

The recommended way is by right dragging the files to the new location. Simply right click on the files you want to move/copy without releasing the mouse button. Then drag the files to the new location and release the mouse button. A context menu will appear where you can either choose Context Menu → SVN Copy versioned files here. or Context Menu → SVN Move versioned files here.

B.2. Kako prisilim uporabnika, da vnese sporočilo dnevniškega zapisa

Obstajata dva načina, kako uporabnikom preprečiti objavljanje s praznim sporočilom dnevniškega zapisa. Eden je poseben za TortoiseSVN, drugi pa je uporaben za vse odjemalce Subversion, vendar morate za nastavitev imeti neposreden dostop do strežnika.

B.2.1. Ukazna datoteka akcije na strežniku

Če imate neposreden dostop do strežnika skladišča, lahko nastavite ukazno datoteko za akcijo pred objavo, ki zavrne vse objave s praznim sporočilom dnevniškega zapisa ali s prekratkim sporočilom.

V mapi skladišča na strežniku se nahaja podmapa z imenom `hooks`, ki vsebuje nekaj primerov ukaznih datotek akcij, ki jih lahko uporabite. Datoteka `pre-commit.tmpl` vsebuje primer, ki zavrne objavo brez sporočila dnevniškega zapisa ali s prekratkim sporočilom. Zraven so zapisana tudi navodila za namestitev/uporabo ukazne datoteke. Sledite jim.

Ta metoda je priporočljiva, če uporabniki poleg TortoiseSVN uporabljajo še druge odjemalce za Subversion. Slaba stran metode je, da objavo zavrne strežnik, zato bodo uporabniki dobili sporočilo o napaki. Odjemalec namreč ne more vnaprej vedeti, da bo objava zavrnjena. Če želite, da TortoiseSVN onemogoči gumb V redu, dokler ni vneseno sporočilo dnevniškega zapisa predpisane dolžine, uporabite metodo, ki je opisana spodaj.

B.2.2. Lasnosti projekta

TortoiseSVN uporablja lastnosti, da kontrolira nekatere zmožnosti programa. Ena takšnih je lastnost `tsvn:logminsize`.

Če nastavite to lastnost na mapi, bo TortoiseSVN onemogočil gumb V redu v vseh oknih objave, dokler uporabnik ne bo vnesel sporočila dnevniškega zapisa predpisane dolžine.

For detailed information on those project properties, please refer to Razdelek 4.18, “Nastavitve projekta”.

B.3. Kako posodobim izbrane datoteke iz skladišča

Običajno posodobite delovno kopijo z ukazom TortoiseSVN → Posodobi. Če pa želite posodobiti le nekaj novih datotek, ki jih je dodal vaš sodelavec, ne želite pa hkrati spajati ostalih datotek, uporabite drugačen pristop.

Uporabite TortoiseSVN → Preveri spremembe in kliknite na gumb Preveri skladišče. S tem vidite, kaj se je spremenilo v skladišču. Izberite datoteke, ki jih želite posodobiti v delovni kopiji, potem pa izvedite posodobitev preko ukazov kontekstnega menija.

B.4. Kako prevrtim nazaj revizije v skladišču

B.4.1. Uporabite okno za prikaz dnevniških zapisov

By far the easiest way to revert the changes from one or more revisions, is to use the revision log dialog.

1. Izberite datoteko ali mapo, kateri želite povrniti spremembe. Če želite povrniti vse spremembe, izberite vrhnjo datoteko.
2. Za seznam revizij izberite TortoiseSVN → Pokaži dnevnik. Mogoče boste morali uporabiti gumb Pokaži vse ali gumb Naslednjih 100, da bi videli revizijo(e), ki vas zanima.
3. Select the revision you wish to revert. If you want to undo a range of revisions, select the first one and hold the **Shift** key while selecting the last one. If you want to pick out individual revisions and ranges, use the **Ctrl** key while selecting revisions. Right click on the selected revision(s), then select Context Menu → Revert changes from this revision.
4. Če želite staro revizijo postaviti za novo revizijo HEAD, kliknite z desnim gumbom na izbrano revizijo in izberite Kontekstni meni → Povrni na to revizijo. S tem boste izbrisali *vse* spremembe, narejene po izbrani reviziji.

Povrnili ste spremembe znotraj delovne kopije. Preverite rezultate in potem objavite spremembe.

B.4.2. Uporabite okno za spajanje

If you want to enter revision numbers as a list, you can use the Merge dialog. The previous method uses merging behind the scenes; this method uses it explicitly.

1. V delovni kopiji izberite TortoiseSVN → Spoji.
2. In the Merge Type dialog select Merge a range of revisions.
3. In the From: field enter the full repository URL of your working copy folder. This should come up as the default URL.
4. In the Revision range to merge field enter the list of revisions to roll back (or use the log dialog to select them as described above).
5. Make sure the Reverse merge checkbox is checked.
6. In the Merge options dialog accept the defaults.
7. Click Merge to complete the merge.

You have reverted the changes within your working copy. Check that the results are as expected, then commit the changes.

B.4.3. Uporabite `svndumpfilter`

Ker TortoiseSVN nikoli ne izgubi podatkov, revizije, ki ste jih povrnili, še vedno obstajajo v skladišču. Vse, kar ste naredili, je, da ste spremenili zadnjo (HEAD) revizijo na vsebino predhodnje revizije. Če želite, da določene revizije izginejo iz skladišča in da se za njimi izgubi vsaka sled, morate uporabiti bolj drastične ukrepe. Razen v primeru zelo dobrega razloga to *ni priporočljivo*. Eden od možnih razlogov za uporabo te metode je objava zaupnega dokumenta v javno skladišče.

The only way to remove data from the repository is to use the Subversion command line tool `svnadmin`. You can find a description of how this works in the *Repository Maintenance* [http://svnbook.red-bean.com/en/1.8/svn.reposadmin.maint.html].

B.5. Compare two revisions of a file or folder

If you want to compare two revisions in an item's history, for example revisions 100 and 200 of the same file, just use TortoiseSVN → Show Log to list the revision history for that file. Pick the two revisions you want to compare then use Context Menu → Compare Revisions.

If you want to compare the same item in two different trees, for example the trunk and a branch, you can use the repository browser to open up both trees, select the file in both places, then use Context Menu → Compare Revisions.

If you want to compare two trees to see what has changed, for example the trunk and a tagged release, you can use TortoiseSVN → Revision Graph Select the two nodes to compare, then use Context Menu → Compare HEAD Revisions. This will show a list of changed files, and you can then select individual files to view the changes in detail. You can also export a tree structure containing all the changed files, or simply a list of all changed files. Read Razdelek 4.11.3, “Primerjanje map” for more information. Alternatively use Context Menu → Unified Diff of HEAD Revisions to see a summary of all differences, with minimal context.

B.6. Kako vključim skupni podprojekt

Sometimes you will want to include another project within your working copy, perhaps some library code. There are at least 4 ways of dealing with this.

B.6.1. Uporabite lastnost svn:externals

Set the `svn:externals` property for a folder in your project. This property consists of one or more lines; each line has the name of a sub-folder which you want to use as the checkout folder for common code, and the repository URL that you want to be checked out there. For full details refer to Razdelek 4.19, “External Items”.

Objavite novo mapo. Pri posodobitvi bo Subversion potegnil kopijo projekta iz skladišča v vašo delovno kopijo. Če je to potrebno, se podmape ustvarijo samodejno. Ob vsaki posodobitvi delovne kopije boste posodobili tudi vse zunanje projekte.

If the external project is in the same repository, any changes you make there will be included in the commit list when you commit your main project.

If the external project is in a different repository, any changes you make to the external project will be shown or indicated when you commit the main project, but you have to commit those external changes separately.

Od vseh treh opisanih možnosti je ta edina, ki ne potrebuje namestitve na strani odjemalca. Ko so zunanje datoteke določene v lastnostih map, bodo ob posodobitvi vsi odjemalci popolnili mape.

B.6.2. Uporabite vgnezdeno delovno kopijo

Create a new folder within your project to contain the common code, but do not add it to Subversion.

Na novi mapi izvedite TortoiseSVN → Prevzem skupne kode. Sedaj imate ločeno delovno kopijo vgnezdeno v svoji glavni delovni kopiji.

Ti dve delovni kopiji sta neodvisni. Ko objavite spremembe na korenski delovni kopiji, se spremembe na vgnezdeni delovni kopij prezrejo. Podobno je v primeru posodobitve: ko posodobite korensko delovno kopijo, je vgnezdena prezrta.

B.6.3. Uporabite relativno lokacijo

If you use the same common core code in several projects, and you do not want to keep multiple working copies of it for every project that uses it, you can just check it out to a separate location which is related to all the other projects which use it. For example:

```
C:\Projects\Proj1
C:\Projects\Proj2
C:\Projects\Proj3
C:\Projects\Common
```

and refer to the common code using a relative path, e.g. `..\..\Common\DSPcore`.

If your projects are scattered in unrelated locations you can use a variant of this, which is to put the common code in one location and use drive letter substitution to map that location to something you can hard code in your projects, e.g. Checkout the common code to `D:\Documents\Framework` or `C:\Documents and Settings\{login}\My Documents\framework` then use

```
SUBST X: "D:\Documents\framework"
```

to create the drive mapping used in your source code. Your code can then use absolute locations.

```
#include "X:\superio\superio.h"
```

Ta metoda deluje le v okoljih, kjer vsi uporabniki uporabljajo računalnike PC. Preslikave pogonov boste morali dokumentirati, da bodo ostali člani teama vedeli, od kje prihajajo te skrivnostne datoteke. Metoda naj se uporablja v zaprtem krogu razvijalcev, za splošno uporabo pa ni priporočljiva.

B.6.4. Add the project to the repository

The maybe easiest way is to simply add the project in a subfolder to your own project working copy. However this has the disadvantage that you have to update and upgrade this external project manually.

To help with the upgrade, TortoiseSVN provides a command in the explorer right-drag context menu. Simply right-drag the folder where you unzipped the new version of the external library to the folder in your working copy, and then select Context Menu → SVN Vendorbranch here. This will then copy the new files over to the target folder while automatically adding new files and removing files that aren't in the new version anymore.

B.7. Kako ustvarim bližnjico do skladišča

If you frequently need to open the repository browser at a particular location, you can create a desktop shortcut using the automation interface to TortoiseProc. Just create a new shortcut and set the target to:

```
TortoiseProc.exe /command:repobrowser /path:"url/to/repository"
```

Of course you need to include the real repository URL.

B.8. Kako dodam na seznam prezrtih datoteke, ki so že pod nadzorom

Če ste pomotoma dodali nekatere datoteke, ki bi jih morali preskočiti, kako jih dobite iz sistema nadzora, ne da bi jih izgubili? Mogoče imate svojo konfiguracijsko datoteko za razvojno okolje, ki ni del projekta, ste pa porabili kar nekaj časa, da ste si okolje nastavili po svojih željah.

If you have not yet committed the add, then all you have to do is use TortoiseSVN → Undo Add... to undo the add. You should then add the file(s) to the ignore list so they don't get added again later by mistake.

If the files are already in the repository, they have to be deleted from the repository and added to the ignore list. Fortunately TortoiseSVN has a convenient shortcut for doing this. TortoiseSVN → Unversion and add to ignore

list will first mark the file/folder for deletion from the repository, keeping the local copy. It also adds this item to the ignore list so that it will not be added back into Subversion again by mistake. Once this is done you just need to commit the parent folder.

B.9. Odstranjevanje delovne kopije iz nadzora različic

If you have a working copy which you want to convert back to a plain folder tree without the `.svn` directory, you can simply export it to itself. Read Razdelek 4.27.1, “Kako odstranim delovno kopijo iz nadzora različic” to find out how.

B.10. Kako odstranim delovno kopijo

If you have a working copy which you no longer need, how do you get rid of it cleanly? Easy - just delete it in Windows Explorer! Working copies are private local entities, and they are self-contained. Deleting a working copy in Windows Explorer does not affect the data in the repository at all.

Dodatek C. Uporabni namigi za skrbnike sistema

Ta dodatek podaja rešitve problemov/vprašanj, ki se lahko pojavijo skrbnikom, ki so zadolženi za namestitev TortoiseSVN na večje število računalnikov.

C.1. Namestitev TortoiseSVN preko pravic skupin

Namestitveni program TortoiseSVN je oblike datoteke MSI, kar pomeni, da ne bi smeli imeti nobenih težav pri dodajanju namestitve v pravice skupin (group policies) domenskega kontrolerja.

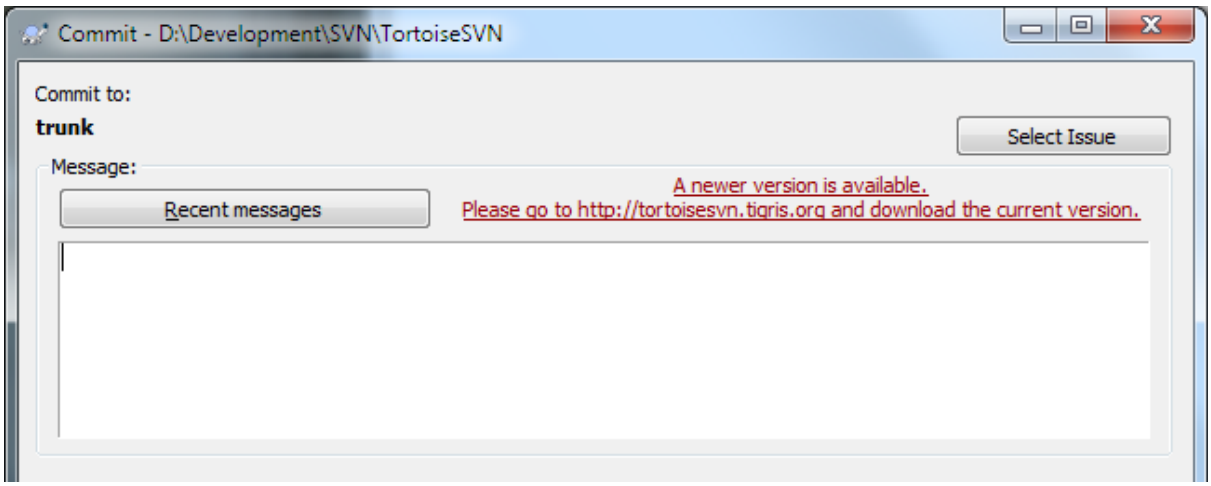
Dober članek na to temo se nahaja v bazi znanja podjetja Microsoft pod številko 314934: http://support.microsoft.com/?kbid=314934.

TortoiseSVN must be installed under *Computer Configuration* and not under *User Configuration*. This is because TortoiseSVN needs the CRT and MFC DLLs, which can only be deployed *per computer* and not *per user*. If you really must install TortoiseSVN on a per user basis, then you must first install the MFC and CRT package version 12 from Microsoft on each computer you want to install TortoiseSVN as per user.

You can customize the MSI file if you wish so that all your users end up with the same settings. TSVN settings are stored in the registry under `HKEY_CURRENT_USER\Software\TortoiseSVN` and general Subversion settings (which affect all Subversion clients) are stored in config files under `%APPDATA%\Subversion`. If you need help with MSI customization, try one of the MSI transform forums or search the web for “MSI transform”.

C.2. Preusmerjanje iskanja najnovejše različice

TortoiseSVN checks if there's a new version available every few days. If there is a newer version available, a notification is shown in the commit dialog.



Slika C.1. The commit dialog, showing the upgrade notification

If you're responsible for a lot of users in your domain, you might want your users to use only versions you have approved and not have them install always the latest version. You probably don't want that upgrade notification to show up so your users don't go and upgrade immediately.

Versions 1.4.0 and later of TortoiseSVN allow you to redirect that upgrade check to your intranet server. You can set the registry key `HKCU\Software\TortoiseSVN\UpdateCheckURL` (string value) to an URL pointing to a text file in your intranet. That text file must have the following format:

```
1.9.1.6000
```

```
A new version of TortoiseSVN is available for you to download!
http://192.168.2.1/downloads/TortoiseSVN-1.9.1.6000-svn-1.9.1.msi
```

The first line in that file is the version string. You must make sure that it matches the exact version string of the TortoiseSVN installation package. The second line is a custom text, shown in the commit dialog. You can write there whatever you want. Just note that the space in the commit dialog is limited. Too long messages will get truncated! The third line is the URL to the new installation package. This URL is opened when the user clicks on the custom message label in the commit dialog. You can also just point the user to a web page instead of the MSI file directly. The URL is opened with the default web browser, so if you specify a web page, that page is opened and shown to the user. If you specify the MSI package, the browser will ask the user to save the MSI file locally.

C.3. Nastavljanje okoljske spremenljivke `SVN_ASP_DOT_NET_HACK`

As of version 1.4.0 and later, the TortoiseSVN installer doesn't provide the user with the option to set the `SVN_ASP_DOT_NET_HACK` environment variable anymore, since that caused many problems and confusion for users who always install *everything* no matter whether they know what it is for.

But the feature is still available in TortoiseSVN and other svn clients. To enable it you have to set the Windows environment variable named `ASPDOTNETHACK` to `1`. Actually, the value of that environment variable doesn't matter: if the variable exists the feature is active.

Pomembno

Please note that this hack is only necessary if you're still using VS.NET2002. All later versions of Visual Studio do *not* require this hack to be activated! So unless you're using that ancient tool, DO NOT USE THIS!

C.4. Onemogočanje kontekstnega menija

Od različice 1.5.0 naprej TortoiseSVN ponuja onemogočanje (pravzaprav skrivanje) elementov v kontekstnem meniju. Ker je to zmožnost, ki se uporablja le z dobrim razlogom, uporabniškega vmesnika ni na voljo, zato je treba nastavitve opraviti neposredno v registru. Uporablja se za onemogočanje določenih ukazov, ki naj jih uporabnik ne bi uporabljal. Vendar upoštevajte, da s tem skrijete le elemente v kontekstnem meniju *Raziskovalca* in da so ukazi še vedno na razpolago na druge načine, n. pr. preko odjemalca za ukazno vrstico ali celo iz drugih pogovornih oken programa TortoiseSVN samega!

Ključa registra, ki vsebujejo informacije o elementih kontekstnega menija, ki jih je potrebno prikazati, sta `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow` in `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Vsak izmed teh vnosov v registru je vrednost `DWORD`, pri kateri vsak bit ustreza določenemu elementu v kontekstnem meniju. Nastavljen bit pomeni, da je ta element v meniju onemogočen.

Vrednost	Vnos v meniju
0x0000000000000001	Prevzemi
0x0000000000000002	Posodobi
0x0000000000000004	Objavi
0x0000000000000008	Dodaj
0x0000000000000010	Povrni
0x0000000000000020	Čiščenje
0x0000000000000040	Reši
0x0000000000000080	Preklop
0x0000000000000100	Uvoz

Vrednost	Vnos v meniju
0x0000000000000200	Izvozi
0x0000000000000400	Ustvari skladišče tu
0x0000000000000800	Veja/Oznaka
0x0000000000001000	Spoji
0x0000000000002000	Izbriši
0x0000000000004000	Preimenuj
0x0000000000008000	Posodobi na revizijo
0x0000000000010000	Razlikuj
0x0000000000020000	Pokaži dnevnik
0x0000000000040000	Uredi spore
0x0000000000080000	Premakni
0x0000000000100000	Preveri spremembe
0x0000000000200000	Prezri
0x0000000000400000	Brskalnik po skladišču
0x0000000000800000	Okrivi
0x0000000001000000	Ustvari popravek
0x0000000002000000	Namesti popravek
0x0000000004000000	Revision graph
0x0000000008000000	Zaklep
0x0000000010000000	Odstrani zaklep
0x0000000020000000	Lastnosti
0x0000000040000000	Razlikuj z URL
0x0000000080000000	Izbriši datoteke brez različic
0x0000000100000000	Merge All
0x0000000200000000	Diff with previous version
0x0000000400000000	Paste
0x0000000800000000	Posodobitev delovne kopije
0x0000001000000000	Diff later
0x0000002000000000	Diff with 'filename'
0x0000004000000000	Unified diff
0x2000000000000000	Nastavitve
0x4000000000000000	Pomoč
0x8000000000000000	O programu

Tabela C.1. Elementi menija in njihove vrednosti

Example: to disable the “Relocate” the “Delete unversioned items” and the “Settings” menu entries, add the values assigned to the entries like this:

```
0x0000000000080000
+ 0x0000000080000000
+ 0x2000000000000000
```

```
= 0x2000000080080000
```

The lower `DWORD` value (`0x80080000`) must then be stored in `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskLow`, the higher `DWORD` value (`0x20000000`) in `HKEY_CURRENT_USER\Software\TortoiseSVN\ContextMenuEntriesMaskHigh`.

Za ponoven prikaz teh elementov izbrišite zgornja vnosa v registru.

Dodatek D. Avtomatizacija TortoiseSVN

Ker je vse ukaze TortoiseSVN možno krmiliti preko parameterov ukazne vrstice, lahko za avtomatizacijo uporabite paketno datoteko ali pa ukaze in pogovorna okna poganjate z določeni parameteri znotraj drugih programov (n. pr. iz vašega priljubljenega urejevalnika besedil).

Pomembno

Upoštevajte, da je TortoiseSVN odjemalec z grafičnim uporabniškim vmesnikom, to poglavje pa vam pove, kako lahko prikažete okna programa TortoiseSVN, tako da od uporabnika pridobite informacije. Če želite napisati ukazno datoteko, ki ne potrebuje vnosa s strani uporabnika, namesto TortoiseSVN uporabite uradni odjemalec Subversion za ukazno vrstico.

D.1. Ukazi TortoiseSVN

Program TortoiseSVN se imenuje `TortoiseProc.exe`. Vse ukaze kličemo s parametrom `/command:abcd`, kjer je `abcd` ime zahtevanega ukaza. Večina ukazov potrebuje vsaj eno pot kot parameter, ki ga podamo z `/path:"neka\pot"`. V spodnji tabeli ukaz ustreza `/command:abcd`, pot pa parametru `/path:"neka\pot"`.

There's a special command that does not require the parameter `/command:abcd` but, if nothing is specified on the command line, starts the project monitor instead. If `/tray` is specified, the project monitor starts hidden and only adds its icon to the system tray.

Ker nekateri ukazi sprejmejo seznam ciljnih poti (n. pr. ko objavljate več datotek), lahko podamo parametru `/path` več poti, ki jih ločimo z znakom `*`.

You can also specify a file which contains a list of paths, separated by newlines. The file must be in UTF-16 format, without a *BOM* [https://en.wikipedia.org/wiki/Byte-order_mark]. If you pass such a file, use `/pathfile` instead of `/path`. To have TortoiseProc delete that file after the command is finished, you can pass the parameter `/deletepathfile`. If you don't pass `/deletepathfile`, you have to delete the file yourself or the file gets left behind.

Pogovorno okno, ki se uporablja pri objavah, posodobitvah in ostalih ukazih običajno ostane odprto dokler ne pritisnemo na gumb V redu. To obnašanje lahko spremenimo z nastavitvami aplikacije. Vendar se bo pogovorno okno zaprlo v vsakem primeru - če ga pokličete iz paketne datoteke ali iz kontekstnega menija.

Za nastavitev druge lokacije konfiguracijske datoteke uporabite parameter `/configdir:"pot\do\konfiguracijske\mape"`. S tem prepišete privzete vrednosti, vključno z zapisi v registru.

To close the progress dialog at the end of a command automatically without using the permanent setting you can pass the `/closeonend` parameter.

- `/closeonend:0` okna ne zapri samodejno
- `/closeonend:1` samodejno zapri, če ni napak
- `/closeonend:2` samodejno zapri, če ni napak in sporov
- `/closeonend:3` samodejno zapri, če ni napak, sporov ali spajanj

To close the progress dialog for local operations if there were no errors or conflicts, pass the `/closeforlocal` parameter.

Spodnja tabela prikazuje vse ukaze, do katerih lahko dostopate preko programa TortoiseProc.exe. Ko smo že omenili zgoraj, uporabljamo obliko `/command:abcd`. V tabeli smo zaradi varčevanja s prostorom opustili predpono `/command`.

Ukaz	Opis
:about	Pokaže pogovorno okno "O programu". Prikaže se tudi, če ne podamo pokaza.

:log	Opens the log dialog. The `/path` specifies the file or folder for which the log should be shown. Additional options can be set:
	• `/startrev:xxx`,
	• `/endrev:xxx`,
	• `/strict` enables the 'stop-on-copy' checkbox,
	• `/merge` enables the 'include merged revisions' checkbox,
	• `/datemin:"{datestring}"` sets the start date of the filter, and
	• `/datemax:"{datestring}"` sets the end date of the filter. The date format is the same as used for svn date revisions.
	• `/findstring:"filterstring"` fills in the filter text,
	• `/findtext` forces the filter to use text, not regex, or
	• `/findregex` forces the filter to use regex, not simple text search, and
	• `/findtype:X` with X being a number between 0 and 511. The numbers are the sum of the following options:
	  • `/findtype:0` filter by everything
	  • `/findtype:1` filter by messages
	  • `/findtype:2` filter by path
	  • `/findtype:4` filter by authors
	  • `/findtype:8` filter by revisions
	  • `/findtype:16` not used
	  • `/findtype:32` filter by bug ID
	  • `/findtype:64` not used
	  • `/findtype:128` filter by date
	  • `/findtype:256` filter by date range
	• If `/outfile:path\to\file` is specified, the selected revisions are written to that file when the log dialog is closed. The revisions are written in the same format as is used to specify revisions in the merge dialog.
	An svn date revision can be in one of the following formats:
	• {2006-02-17}
	• {15:30}
	• {15:30:00.200000}
	• {"2006-02-17 15:30"}

Ukaz	Opis
	• {"2006-02-17 15:30 +0230"} • {2006-02-17T15:30} • {2006-02-17T15:30Z} • {2006-02-17T15:30-04:00} • {20060217T1530} • {20060217T1530Z} • {20060217T1530-0500}
:checkout	Opens the checkout dialog. The `/path` specifies the target directory and the `/url` specifies the URL to checkout from. If you specify the key `/blockpathadjustments`, the automatic checkout path adjustments are blocked. The `/revision:XXX` specifies the revision to check out.
:import	Opens the import dialog. The `/path` specifies the directory with the data to import. You can also specify the `/logmsg` switch to pass a predefined log message to the import dialog. Or, if you don't want to pass the log message on the command line, use `/logmsgfile:pot`, where `pot` points to a file containing the log message.
:update	Updates the working copy in `/path` to HEAD. If the option `/rev` is given then a dialog is shown to ask the user to which revision the update should go. To avoid the dialog specify a revision number `/rev:1234`. Other options are `/nonrecursive`, `/ignoreexternals` and `/ignoreexternals`. The `/stickydepth` indicates that the specified depth should be sticky, creating a sparse checkout. The `/skipprechecks` can be set to skip all checks that are done before an update. If this is specified, then the Show log button is disabled, and the context menu to show diffs is also disabled after the update.
:commit	Odpre pogovorno okno za objave. `/path` določa ciljno mapo ali seznam datotek za objavo. Določite lahko tudi stikalo `/logmsg`, s katerim določite sporočilo dnevniškega zapisa. Če pa ne želite podati sporočila v ukazni vrstici, uporabite `/logmsgfile:pot`, kjer `pot` kaže na datoteko, ki vsebuje sporočilo zapisa. Za izpolnitev polja ID hrošča (če ste pravilno nastavili integracijo s sledilnikom zadev) lahko uporabite `/bugid:"številka hrošča"`.
:add	Doda datoteke v `/path` v sistem verzioniranja.
:revert	Povrne krajevne spremembe v delovni kopiji. `/path` pove, katere elemente je potrebno povrniti.
:cleanup	Cleans up interrupted or aborted operations and unlocks the working copy in `/path`. You also have to pass the `/cleanup` to actually do the cleanup. Use `/noui` to prevent the result dialog from popping up (either telling about the cleanup being finished or showing an error message). `/noprogressui` also disables the progress dialog. `/nodlg` disables showing the cleanup dialog where the user can choose what exactly should be done in the cleanup. The available actions can be specified with the options `/cleanup` for status cleanup, `/breaklocks` to break all locks, `/revert` to revert uncommitted changes, `/delunversioned`, `/delignored`, `/refreshshell`, `/externals`, `/fixtimestamps` and `/vacuum`.
:resolve	Označi sporno datoteko, podano v `/path`, kot rešeno. Če je podana možnost `/noquestion`, se označevanje izvede brez uporanikove potrditve.
:repocreate	V `/path` ustvari skladišče
:switch	Opens the switch dialog. The `/path` specifies the target directory and `/url` the URL to switch to.

Ukaz	Opis
:export	Exports the working copy in `/path` to another directory. If the `/path` points to an unversioned directory, a dialog will ask for an URL to export to the directory in `/path`. If you specify the key `/blockpathadjustments`, the automatic export path adjustments are blocked.
:dropexport	Exports the working copy in `/path` to the directory specified in `/droptarget`. This exporting does not use the export dialog but executes directly. The option `/overwrite` specifies that existing files are overwritten without user confirmation, and the option `/autorename` specifies that if files already exist, the exported files get automatically renamed to avoid overwriting them. The option `/extended` can specify either `localchanges` to only export files that got changed locally, or `unversioned` to also export all unversioned items as well.
:dropvendor	Copies the folder in `/path` recursively to the directory specified in `/droptarget`. New files are added automatically, and missing files get removed in the target working copy, basically ensuring that source and destination are exactly the same. Specify `/noui` to skip the confirmation dialog, and `/noprogressui` to also disable showing the progress dialog.
:merge	Opens the merge dialog. The `/path` specifies the target directory. For merging a revision range, the following options are available: `/fromurl:URL`, `/revrange:string`. For merging two repository trees, the following options are available: `/fromurl:URL`, `/tourl:URL`, `/fromrev:xxx` and `/torev:xxx`.
:mergeall	Odpre okno za spajanje. `/path` določa ciljno mapo.
:copy	Brings up the branch/tag dialog. The `/path` is the working copy to branch/tag from. And the `/url` is the target URL. If the urls starts with a ^ it is assumed to be relative to the repository root. To already check the option `Switch working copy to new branch/tag` you can pass the `/switchaftercopy` switch. To check the option `Create intermediate folders` pass the `/makeparents` switch. You can also specify the `/logmsg` switch to pass a predefined log message to the branch/tag dialog. Or, if you don't want to pass the log message on the command line, use `/logmsgfile:pot`, where `pot` points to a file containing the log message.
:settings	Odpre okno za nastavitve
:remove	Odstrani datoteke v `/path` iz sistema Subversion.
:rename	Preimenuje datoteko v `/path`. Novo ime vpišete v pogovornem oknu. Če se želite izgoniti vprašanju o preimenovanju podobnih datotek v enem koraku, podajte `/noquestion`.
:diff	Starts the external diff program specified in the TortoiseSVN settings. The `/path` specifies the first file. If the option `/path2` is set, then the diff program is started with those two files. If `/path2` is omitted, then the diff is done between the file in `/path` and its BASE. If the specified file also has property modifications, the external diff tool is also started for each modified property. To prevent that, pass the option `/ignoreprops`. To explicitly set the revision numbers use `/startrev:xxx` and `/endrev:xxx`, and for the optional peg revision use `/pegrevision:xxx`. If `/blame` is set and `/path2` is not set, then the diff is done by first blaming the files with the given revisions. The parameter `/line:xxx` specifies the line to jump to when the diff is shown.
:showcompare	Odvisno od naslovov URL in revizij, ki jih želite primerjati, se prikaže poenotena razlika (če je nastavljena možnost `unified`), seznam spremenjenih datotek ali (v primeru, da naslov URL kaže na datoteko) pregledovalnik sprememb za ti dve datoteki.

Ukaz	Opis
	Možnosti `url1`, `url2`, `revision1` in `revision2` so obvezne, možnosti `pegrevision`, `ignoreancestry`, `blame` in `unified` pa opcijske. If the specified url also has property modifications, the external diff tool is also started for each modified property. To prevent that, pass the option `/ignoreprops`.
:conflicteditor	Zažene urejevalnik sporov, ki je naveden v nastavitvah TortoiseSVN, s pravilnimi datotekami za sporno datoteko `/path`.
:relocate	Odpre pogovorno okno za premestitev. `/path` določa pot delovne kopije za premestitev.
:help	Odpre datoteko pomoči.
:repostatus	Opens the check-for-modifications dialog. The `/path` specifies the working copy directory. If `/remote` is specified, the dialog contacts the repository immediately on startup, as if the user clicked on the Check repository button.
:repobrowser	Starts the repository browser dialog, pointing to the URL of the working copy given in `/path` or `/path` points directly to an URL. An additional option `/rev:xxx` can be used to specify the revision which the repository browser should show. If the `/rev:xxx` is omitted, it defaults to HEAD. If `/path` points to an URL, the `/projectpropertiespath:path/to/wc` specifies the path from where to read and use the project properties. If `/outfile:path\to\file` is specified, the selected URL and revision are written to that file when the repository browser is closed. The first line in that text file contains the URL, the second line the revision in text format.
:ignore	Doda vse cilje v `/path` na seznam prezrtih elementov, to pomeni, da datotekam doda lastnost `svn:ignore`.
:blame	Odpre pogovorno okno Okrivi za datoteko, podano z opcijo `/path`. Če sta nastavljevni možnosti `/startrev` in `/endrev`, se ne pojavi pogovorno okno za vnos območja revizij, ampak se uporabita navedeni vrednosti. Če je nastavljena opcija `/line:nnn`, se ob zagonu TortoiseBlame prikaže navedena vrstica. Podprte so tudi možnosti `/ignoreeol`, `/ignorespaces` in `/ignoreallspaces`.
:cat	Shrani datoteko z naslova URL ali delovne kopije, navedene v `/path`, na lokacijo `/savepath:pot`. `/revision:xxx` predstavlja številko revizije. S tem lahko prevzamemo datoteko v določeni reviziji.
:createpatch	Creates a patch file for the path given in `/path`. To skip the file Save-As dialog you can pass `/savepath:pot` to specify the path where to save the patch file to directly. To prevent the unified diff viewer from being started showing the patch file, pass `/noview`.
:revisiongraph	Shows the revision graph for the path given in `/path`. To create an image file of the revision graph for a specific path, but without showing the graph window, pass `/output:path` with the path to the output file. The output file must have an extension that the revision graph can actually export to. These are: `.svg`, `.wmf`, `.png`, `.jpg`, `.bmp` and `.gif`. Since the revision graph has many options that affect how it is shown, you can also set the options to use when creating the output image file. Pass these options

Ukaz	Opis
	with `/options:XXXX`, where `XXXX` is a decimal value. The best way to find the required options is to start the revision graph the usual way, set all user-interface options and close the graph. Then the options you need to pass on the command line can be read from the registry `HKCU\Software\TortoiseSVN\RevisionGraphOptions`.
:lock	Zaklene datoteko ali vse datoteke v mapi, podani v `/path`. Pogovorno okno za zaklepanje se pojavi, da lahko uporabnik vnese sporočilo zaklepanja.
:unlock	Odklene datoteko ali vse datoteke v mapi, podani v `/path`.
:rebuildiconcache	Ponovno zgradi predpomnilnik ikon sistema. Ta ukaz uporabite le v primeru, če se ikone na nek način poškodujejo. Neizogiben stranski učinek tega postopka je, da se ikone na namizju prerazporedijo po svoje. Če sporočila ne želite videti, podajte `/noquestion`.
:properties	Shows the properties dialog for the path given in `/path`. For dealing with versioned properties this command requires a working copy. Revision properties can be viewed/changed if `/path` is an URL and `/rev:XXX` is specified. To open the properties dialog directly for a specific property, pass the property name as `/property:name`.
:sync	Exports/imports settings, either depending on whether the current settings or the exported settings are newer, or as specified. If a path is passed with `/path`, then the path is used to store or read the settings from. The parameter `/askforpath` will show a file open/save dialog for the user to chose the export/import path. If neither `/load` nor `/save` is specified, then TortoiseSVN determines whether to export or import the settings by looking at which ones are more recent. If the export file is more recent than the current settings, then the settings are loaded from the file. If the current settings are more recent, then the settings are exported to the settings file. If `/load` is specified, the settings are imported from the settings file. If `/save` is specified, the current settings are exported to the settings file. The parameter `/local` forces a settings export to include local settings, i.e. settings that refer to local paths.

Tabela D.1. Seznam ukazov in možnosti

Primeri (vpisati jih je potrebno v eno vrstico):

```
TortoiseProc.exe /command:commit
                 /path:"c:\svn_wc\file1.txt*c:\svn_wc\file2.txt"
                 /logmsg:"sporočilo" /closeonend:0

TortoiseProc.exe /command:update /path:"c:\svn_wc\" /closeonend:0

TortoiseProc.exe /command:log /path:"c:\svn_wc\file1.txt"
                 /startrev:50 /endrev:60 /closeonend:0
```

D.2. Tsvncmd URL handler

Using special URLs, it is also possible to call TortoiseProc from a web page.

TortoiseSVN registers a new protocol `tsvncmd:` which can be used to create hyperlinks that execute TortoiseSVN commands. The commands and parameters are the same as when automating TortoiseSVN from the command line.

The format of the `tsvncmd:` URL is like this:

```
tsvncmd:command:cmd?parameter:paramvalue?parameter:paramvalue
```

with `cmd` being one of the allowed commands, `parameter` being the name of a parameter like `path` or `revision`, and `paramvalue` being the value to use for that parameter. The list of parameters allowed depends on the command used.

The following commands are allowed with `tsvncmd:` URLs:

- `:update`
- `:commit`
- `:diff`
- `:repobrowser`
- `:checkout`
- `:export`
- `:blame`
- `:repostatus`
- `:revisiongraph`
- `:showcompare`
- `:log`

A simple example URL might look like this:

```
<a href="tsvncmd:command:update?path:c:\svn_wc?rev:1234">Update</a>
```

or in a more complex case:

```
<a href="tsvncmd:command:showcompare?
url1:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
url2:https://svn.code.sf.net/p/stefanstools/code/trunk/StExBar/src/setup/Setup.wxs?
revision1:188?revision2:189">compare</a>
```

D.3. Ukazi TortoiseIDiff

Orodje za razlikovanje slik omogoča nekaj možnosti ukazne vrstice, s katerimi lahko določite, kako se orodje zažene. Program se imenuje `TortoiseIDiff.exe`.

Spodnja tabela vsebuje seznam vseh možnosti, ki jih lahko v ukazni vrstici podate orodju za razlikovanje slik.

Možnost	Opis
:left	Pot do datoteke, prikazane na levi.

Možnost	Opis
:lefttitle	Naslov. Uporablja se pri prikazu slike namesto polne poti datoteke.
:right	Pot do datoteke, prikazane na desni.
:righttitle	Naslov. Uporablja se pri prikazu slike namesto polne poti datoteke.
:overlay	Če je parameter podan, orodje za razlikovanje slik uporablja prekrivni način (alpha blend).
:fit	Če je parameter podan, orodje za razlikovanje slik stlači sliki v okno.
:showinfo	Pokaže okno z informacijami o sliki.

Tabela D.2. Seznam možnosti

Primer (vpisati ga je potrebno v eni vrstici):

```
TortoiseIDiff.exe /left:"c:\images\img1.jpg" /lefttitle:"image 1"
                  /right:"c:\images\img2.jpg" /righttitle:"image 2"
                  /fit /overlay
```

D.4. TortoiseUDiff Commands

The unified diff viewer has only two command line options:

Možnost	Opis
:patchfile	Path to the unified diff file.
:p	Activates pipe mode. The unified diff is read from the console input.

Tabela D.3. Seznam možnosti

Examples (which should be entered on one line):

```
TortoiseUDiff.exe /patchfile:"c:\diff.patch"
```

If you create the diff from another command, you can use TortoiseUDiff to show that diff directly:

```
svn diff | TortoiseUDiff.exe /u
```

this also works if you omit the `/p` parameter:

```
svn diff | TortoiseUDiff.exe
```

Dodatek E. Ustrezni ukazi v odjemalcu za ukazno vrstico

Ta priročnik vas včasih pozove, da pogledate dokumentacijo sistema Subversion, ki opisuje sistem Subversion z vidika uporabe preko ukazne vrstice (Command Line Interface - CLI). Da bi lažje razumeli, kaj se dogaja v ozadju, je tu seznam ukazov TortoiseSVN in ekvivalentnih ukazov odjemalca za ukazno vrstico.

> **Opomba**
>
> Čeprav za ukaze TortoiseSVN obstajajo ekvivalentni ukazi v ukazni vrstici, ne pozabite, da TortoiseSVN ukazne vrstice *ne* uporablja, ampak neposredno uporablja knjižnice sistema Subversion.

Če mislite, da ste našli napako v programu TortoiseSVN, vas bomo mogoče prosili, da jo poskusite ponoviti z uporabo odjemalca ukazne vrstice za Subversion. S tem ločimo napake programa TortoiseSVN od napak programa Subversion. Ta razpredelnica vam pove, katere ukaze poskusite.

E.1. Konvencije in osnovna pravila

In the descriptions which follow, the URL for a repository location is shown simply as `URL`, and an example might be `https://svn.code.sf.net/p/tortoisesvn/code/trunk/`. The working copy path is shown simply as `PATH`, and an example might be `C:\TortoiseSVN\trunk`.

> **Pomembno**
>
> Ker je TortoiseSVN razširitev lupine Windows, ne more uporabljati trenutne mape. Vse poti delovne kopije morajo biti podane z absolutno in ne z relativno potjo.

Določeni elementi so opcijski in se ponavadi v TortoiseSVN nastavijo preko potrditvenih polj ali radijskih gumbov. Te opcije so v definicijah ukazne vrstice prikazane v [oglatih oklepajih].

E.2. Ukazi TortoiseSVN

E.2.1. Prevzemi

```
svn checkout [-depth ARG] [--ignore-externals] [-r rev] URL PATH
```

The depth combo box items relate to the `-depth` argument.

Če je potrjeno polje Izpusti zunanje, uporabi stikalo `--ignore-externals`.

Če prevzemate določeno revizijo, pove to po navedbi naslova URL s stikalom `-r`.

E.2.2. Posodobi

```
svn info URL_of_WC
svn update [-r rev] PATH
```

Posodabljanje večjega števila elementov v sistemu Subversion ni atomična operacija. TortoiseSVN najprej najde revizijo HEAD v skladišču in potem posodobi vse elemente na to določeno številko revizije, da se izogne ustarjanju mešane delovne kopije.

Če je za posodobitev izbran le en element ali če niso vsi izbrani elementi iz istega skladišča, TortoiseSVN posodobi na revizijo HEAD.

Tu se ne uporabljajo možnosti ukazne vrstice. Posodobi na revizijo prav tako izvede posodabljanje, a ponudi več možnosti.

E.2.3. Posodobi na revizijo

```
svn info URL_of_WC
svn update [-r rev] [-depth ARG] [--ignore-externals] PATH
```

The depth combo box items relate to the `-depth` argument.

Če je potrjeno polje Izpusti zunanje, uporabi stikalo `--ignore-externals`.

E.2.4. Objavi

V programu TortoiseSVN pogovorno okno za objavo uporablja več ukazov sistema Subversion. Prvi ukaz je preverjanje stanja, ki v delovni kopiji najde elemente, ki so kandidati za objavo. Seznam lahko pregledate, pogledate spremembe datotek glede na revizijo BASE in izberete elemente, ki jih želite objaviti.

```
svn status -v PATH
```

Če je potrjena možnost Prikaži datoteke brez različic, bo TortoiseSVN prikazal vse datoteke in mape brez različic v hierarhiji delovne kopije, pri tem pa upošteval pravila za prezrte elemente. Ta zmožnost nima neposrednega ekvivalentnega ukaza v sistemu Subversion, ker se ukaz `svn status` ne ukvarja z datotekami brez različic.

Če potrdite datoteke in mape brez različic, se le-te najprej dodajo v delovno kopijo.

```
svn add PATH...
```

Ko kliknete na gumb V redu, se spremembe objavijo. Če ste pustili vsa potrditvena polja za izbiro datotek v privzetem stanju, TortoiseSVN uporabi enojno rekurzivno objavo delovne kopije. Če nekatere datoteke izločite, mora uporabiti nerekurzivno objavo (`-N`). Vsaka pot mora biti posamično navedena v ukazni vrstici ob objavi.

```
svn commit -m "LogMessage" [-depth ARG] [--no-unlock] PATH...
```

`Sporočilo` predstavlja vsebino vnosnega polja sporočila dnevniškega zapisa. Lahko je prazno.

Če je potrjena možnost Ohrani zaklepe, uporabi stikalo `--no-unlock`.

E.2.5. Razlikuj

```
svn diff PATH
```

Če uporabljate ukaz Razlikuj iz glavnega kontekstnega menija, gledate spremembe v primerjavi z revizijo BASE. To se zgodi tudi pri uporabi ukazne vrstice, izhod pa je podan v obliki poenotene razlike. Vendar za prikaz razlik med dvema datotekama TortoiseSVN ne uporablja te možnosti, ampak TortoiseMerge (ali kak drug program za razlikovanje, ki ste ga izbrali), torej ekvivalentnega ukaza za ukazno vrstico ni.

S TortoiseSVN lahko razlikujete katerikoli dve datoteki, ne glede na to, ali sta pod nadzorom različic ali ne. TortoiseSVN enostavno pošlje datoteki v izbrani program za razlikovanje in mu prepusti, da vam pokaže razlike v datotekah.

E.2.6. Pokaži dnevnik

```
svn log -v -r 0:N --limit 100 [--stop-on-copy] PATH
or
svn log -v -r M:N [--stop-on-copy] PATH
```

Po privezetih nastavitvah TortoiseSVN poskuša pridobiti 100 dnevniških zapisov z uporabo parametra --limit. Če nastavitve zahtevajo uporabo starega aplikacijskega vmesnika, potem se za pridobivanje 100 revizij dnevniških zapisov iz skladišča uporablja druga oblika.

Če je potrjeno polje Ustavi ob kopiranju/preimenovanju, uporabi stikalo `--stop-on-copy`.

E.2.7. Preveri posodobitve

```
svn status -v PATH
or
svn status -u -v PATH
```

Najprej se stanje preveri le v delovni kopiji. Če pa kliknete na Preveri skladišče, potem se preveri tudi stanje v skladišču, da se izve, katere datoteke bi se spremenile v primeru posodobitve. To zahteva stikalo `-u`.

Če je potrjena možnost Prikaži datoteke brez različic, bo TortoiseSVN prikazal vse datoteke in mape brez različic v hierarhiji delovne kopije, pri tem pa upošteval pravila za prezrte elemente. Ta zmožnost nima neposrednega ekvivalentnega ukaza v sistemu Subversion, ker se ukaz `svn status` ne ukvarja z datotekami brez različic.

E.2.8. Graf revizij

Graf revizij je zmožnost aplikacije TortoiseSVN. Ekvivalenten ukaz v ukazni vrstici ne obstaja.

What TortoiseSVN does is an

```
svn info URL_of_WC
svn log -v URL
```

where URL is the repository *root* and then analyzes the data returned.

E.2.9. Brskalnik po skladišču

```
svn info URL_of_WC
svn list [-r rev] -v URL
```

Da najdete korensko mapo skladišča, lahko uporabite ukaz `svn info`. Ta mapa se prikaže kot vrhnja mapa v brskalniku po skladišču. `Višje` od te mape ne morete. Ta ukaz vrne tudi vse informacije o zaklepih, ki se prav tako prikažejo v brskalniku po skladišču.

Ukaz `svn list` s parametrom URL in številko revizije vrne vsebino mape.

E.2.10. Uredi spore

Ekvivalenten ukaz za ukazno vrstico ne obstaja. Uporabi se orodje TortoiseMerge ali kakšen drug tristranski program za razlikovanje/spajanje, s pomočjo katerega se odločimo, kako bomo spore rešili.

E.2.11. Rešeno

```
svn resolved PATH
```

E.2.12. Preimenuj

```
svn rename CURR_PATH NEW_PATH
```

E.2.13. Izbriši

```
svn delete PATH
```

E.2.14. Povrni

```
svn status -v PATH
```

Prva faza predstavlja preverjanje stanja, s čemer ugotovimo, katere elemente v delovni kopiji je mogoče povrniti. Seznam lahko pregledate, pogledate razlike v primerjavi z revizijo BASE in izberete elemente, ki jih želite povrniti.

Ko kliknete na gumb V redu, se zgodi povrnitev. Če izbire niste spreminjali, uporabi TortoiseSVN eno samo rekurzivno (`-R`) povrnitev delovne kopije. Če ste izklopili nekatere datoteke, potem mora pri uporabi ukaza v ukazni vrstici podati vsako pot posebej.

```
svn revert [-R] PATH...
```

E.2.15. Čiščenje

```
svn cleanup PATH
```

E.2.16. Dobi zaklep

```
svn status -v PATH
```

Prva faza je preverjanje stanja, ki ugotovi, katere datoteke v delovni kopiji lahko zaklenete. Nato izberete datoteke, ki jih želite zakleniti.

```
svn lock -m "LockMessage" [--force] PATH...
```

`Sporočilo` predstavlja vsebina sporočila zaklepa. Lahko je prazno.

Če je potrjeno polje Ukradi zaklep, uporabi stikalo `--force`.

E.2.17. Odstrani zaklep

```
svn unlock PATH
```

E.2.18. Veja/Oznaka

```
svn copy -m "LogMessage" URL URL
or
svn copy -m "LogMessage" URL@rev URL@rev
or
svn copy -m "LogMessage" PATH URL
```

Okno Veja/Oznaka izdela kopijo skladišča. Na voljo so trije radijski gumbi:

- Revizija HEAD v skladišču
- Določena revizija v skladišču
- Delovna kopija

Ti ustrezajo trem opisanim variacijam ukazne vrstice.

`Sporočilo` predstavlja vsebino vnosnega polja sporočila dnevniškega zapisa. Lahko je prazno.

E.2.19. Preklop

```
svn info URL_of_WC
svn switch [-r rev] URL PATH
```

E.2.20. Spoji

```
svn merge [--dry-run] --force From_URL@revN To_URL@revM PATH
```

The Test Merge performs the same merge with the `--dry-run` switch.

```
svn diff From_URL@revN To_URL@revM
```

Gumb Poenotena različica pokaže operacijo razlikovanja, ki bo uporabljena za spajanje.

E.2.21. Izvozi

```
svn export [-r rev] [--ignore-externals] URL Export_PATH
```

Ta oblika se uporablja v mapi brez različic, kadar je ta mapa tudi ciljna mapa.

Izvoz delovne kopije na drugo lokacijo se ne izvede s pomočjo knjižnice Subversion, zato ekvivalentnega ukaza ni.

TortoiseSVN prekopira vse datoteke na novo lokacijo, hkrati pa prikazuje napredek operacije. Nastavite lahko, da se izvozijo tudi datoteke/mape brez različic.

Če je potrjeno potrditveno polje Izpusti zunanje, v obeh primerih uporabi stikalo `--ignore-externals`.

E.2.22. Premakni

```
svn switch --relocate From_URL To_URL
```

E.2.23. Tu ustvari skladišče

```
svnadmin create --fs-type fsfs PATH
```

E.2.24. Dodaj

```
svn add PATH...
```

Če izberete mapo, jo TortoiseSVN najprej rekurzivno pregleda, da najde elemente, ki jih je možno dodati.

E.2.25. Uvoz

```
svn import -m LogMessage PATH URL
```

`Sporočilo` predstavlja vsebino vnosnega polja sporočila dnevniškega zapisa. Lahko je prazno.

E.2.26. Okrivi

```
svn blame -r N:M -v PATH
svn log -r N:M PATH
```

Če za pregled krivdnih informacij uporabljate TortoiseBlame, potem potrebujete tudi informacije o dnevniških zapisih datoteke, ki se prikažejo kot namig. Če krivdne informacije pregledujete v obliki besedilne datoteke, to ni potrebno.

E.2.27. Dodaj na seznam prezrtih

```
svn propget svn:ignore PATH > tempfile
{edit new ignore item into tempfile}
svn propset svn:ignore -F tempfile PATH
```

Ker lastnost `svn:ignore` pogosto obsega več vrstic, tukaj prikazujemo, kako jo spreminjamo preko besedilne datoteke namesto neposredno v ukazni vrstici.

E.2.28. Ustvari popravek

```
svn diff PATH > patch-file
```

TortoiseSVN ustvari datoteko popravkov v obliki poenotene razlike s primerjanjem delovne kopije in verzije BASE.

E.2.29. Namesti popravek

Nameščanje datotek popravkov je zahtevno opravilo, razen če imata datoteka popravkov in delovna kopija isto številko revizije. Na srečo lahko uporabite orodje TortoiseMerge, ki nima ekvivalentnega orodja v sistemu Subversion.

Dodatek F. Podrobnosti o izvedbi

Ta dodatek vsebuje podrobno razlago izvedbe nekaterih zmožnosti programa TortoiseSVN.

F.1. Prekrivne ikone

Vsaka datoteka in mapa pod nadzorom ima v sistemu Subversion stanje, ki ga sporoči knjižnica Subversion. V odjemalcu za ukazno vrstico je status prikazan z črko, v programu TortoiseSVN pa je prikazan grafično s prekrivnimi ikonami. Ker je število prekrivnih ikon zelo omejeno, lahko vsaka ikona predstavlja več vrednosti stanja.

Prekrivna ikona za *spor* se uporablja za prikaz `spornega stanja`. Pri posodobitvi ali preklopu se pri spajanju krajevnih sprememb in sprememb iz skladišča pojavijo spori. Uporablja se tudi za prikaz `oviranega` stanja, ki se pojavi, ko operacije ni mogoče zaključiti.

Prekrivna ikona *spremenjeno* predstavlja `spremenjeno` stanje pri krajevnih spremembah, `spojeno` stanje, ko so bile spremembe iz skladišča spojene s krajevnimi spremembami in `zamenjano` stanje, ko je bila datoteka izbrisana in nadomeščena z drugo datoteko z istim imenom.

Prekrivna ikona *izbrisano* predstavlja `izbrisano` stanje, kjer je element označen za brisanje ali stanje `manjkajoče`, ko elementa ni v delovni kopiji. Seveda element, ki ne obstaja, ne more imeti prekrivne ikone, zato je ikona za manjkajoč element pripeta na nadrejeno mapo.

Prekrivna ikona *dodano* predstavlja `dodano` stanje, ko je element dodan sistemu za nadzor različic.

Prekrivna ikona *V sistemu Subversion* predstavlja `običajno` stanje ali element pod nadzorom, katerega stanje še ni znano. Ker TortoiseSVN uporablja poseben proces za predpomnenje stanja, lahko traja nekaj sekund, preden se prekrivna ikona posodobi.

The *Needs Lock* overlay is used to indicate when a file has the `svn:needs-lock` property set.

Prekrivna ikona *zaklenjeno* se uporablja, kadar je delovna kopija lastnica zaklepa za datoteko.

Ikona za *prezrte* elemente se uporablja za elemente, ki so `prezrti` zaradi splošnega vzorca prezrtih elementov ali zaradi nastavljene lastnosti `svn:ignore` na nadrejeni mapi. Prikaz je možno izključiti.

Ta ikona se uporablja za elemente `brez različic`. To so elementi, ki se nahajajo znotraj mape pod nadzorom, sami pa niso pod nadzorom. Prikaz te ikone je možno izključiti.

If an item has Subversion status `none` (the item is not within a working copy) then no overlay is shown. If you have chosen to disable the *Ignored* and *Unversioned* overlays then no overlay will be shown for those files either.

Element ima lahko le eno stanje sistema Subversion. Primer: datoteka je krajevno spremenjena in hkrati označena za brisanje. Subversion vrne eno samo stanje - v tem primeru `izbrisano`. Prioritete stanj definira Subversion.

Kadar TortoiseSVN prikazuje stanje rekurzivno (privzeta nastavitev), vsaka mapa prikazuje svoje stanje in stanje vseh svojih elementov. Da prikažemo le eno vrednost stanja - prekrivno ikono za *povzetek* - uporabljamo prioritetni red, ki je prikazan zgoraj. Najvišo prioriteto ima stanje *sporno*.

In fact, you may find that not all of these icons are used on your system. This is because the number of overlays allowed by Windows is limited to 15. Windows uses 4 of those, and the remaining 11 can be used by other applications. If there are not enough overlay slots available, TortoiseSVN tries to be a *Good Citizen (TM)* and limits its use of overlays to give other apps a chance.

Since there are Tortoise clients available for other version control systems, we've created a shared component which is responsible for showing the overlay icons. The technical details are not important here, all you need to know is that this shared component allows all Tortoise clients to use the same overlays and therefore the limit of 11 available slots isn't used up by installing more than one Tortoise client. Of course there's one small drawback: all Tortoise clients use the same overlay icons, so you can't figure out by the overlay icons what version control system a working copy is using.

- Ikone *Navadno*, *Spremenjeno* in *Sporno* se vedno naložijo in so vidne.
- Ikona *Izbrisano* se naloži, če je to mogoče, če pa prostih mest ni dovolj, se prikaže ikona *Spremenjeno*.
- Ikona *Samo za branje* se naloži, če je to mogoče, če pa prostih mest ni dovolj, se prikaže ikona *Običajno*.
- *Locked* is loaded if possible, but falls back to *Normal* if there are not enough slots.
- *Added* is loaded if possible, but falls back to *Modified* if there are not enough slots.

Dodatek G. Language Packs and Spell Checkers

The standard installer has support only for English, but you can download separate language packs and spell check dictionaries separately after installation.

G.1. Jezikovni paketi

The TortoiseSVN user interface has been translated into many different languages, so you may be able to download a language pack to suit your needs. You can find the language packs on our *translation status page* [https://tortoisesvn.net/translation_status_dev.html]. And if there is no language pack available, why not join the team and submit your own translation ;-)

Each language pack is packaged as a `.msi` installer. Just run the install program and follow the instructions. After the installation finishes, the translation will be available.

The documentation has also been translated into several different languages. You can download translated manuals from the *support page* [https://tortoisesvn.net/support.html] on our website.

G.2. Črkovalnik

TortoiseSVN uses the Windows spell checker if it's available (Windows 8 or later). Which means that if you want the spell checker to work in a different language than the default OS language, you have to install the spell checker module in the Windows settings (`Settings > Time & Language > Region & Language`).

TortoiseSVN will use that spell checker if properly configured with the `tsvn:projectlanguage` project property.

In case the Windows spell checker is not available, TortoiseSVN can also use spell checker dictionaries from *OpenOffice* [https://openoffice.org] and *Mozilla* [https://mozilla.org].

The installer automatically adds the US and UK English dictionaries. If you want other languages, the easiest option is simply to install one of TortoiseSVN's language packs. This will install the appropriate dictionary files as well as the TortoiseSVN local user interface. After the installation finishes, the dictionary will be available too.

Or you can install the dictionaries yourself. If you have OpenOffice or Mozilla installed, you can copy those dictionaries, which are located in the installation folders for those applications. Otherwise, you need to download the required dictionary files from http://wiki.services.openoffice.org/wiki/Dictionaries.

Once you have got the dictionary files, you probably need to rename them so that the filenames only have the locale chars in it. Example:

- `en_US.aff`
- `en_US.dic`

Then just copy them into the `%APPDATA%\TortoiseSVN\dic` folder. If that folder isn't there, you have to create it first. TortoiseSVN will also search the `Languages` sub-folder of the TortoiseSVN installation folder (normally this will be `C:\Program Files\TortoiseSVN\Languages`); this is the place where the language packs put their files. However, the %APPDATA%-folder doesn't require administrator privileges and, thus, has higher priority. The next time you start TortoiseSVN, the spell checker will be available.

Če namestite več slovarjev, TortoiseSVN uporabi naslednja pravila, da ugotovi, katerega naj uporabi.

1. Preveri nastavitev lastnosti `tsvn:projectlanguage`. Preberite Razdelek 4.18, “Nastavitve projekta” za več informacij o nastavitvi projektnih lastnosti.
2. Če projektni jezik ni nastavljen ali nastavljeni jezik ni nameščen, poiskusi z jezikom sistema Windows.

3. If the exact Windows locale doesn't work, try the “Base” language, e.g. `de_CH` (Swiss-German) falls back to `de_DE` (German).

4. Če po zgornjih pravilih ne uspe najti slovarja, potem je privzeti jezik angleščina, ki se nahaja v standarnem namestitvenem paketu.

Slovar

Čiščenje	Citat iz knjige The Subversion Book: “ Rekurzivno počisti delovno kopijo, tako da odstrani zaklepe in zaključi nedokončane operacije. Če se vam kdaj prikaže sporočilo *delovna kopija zaklenjena*, poženite ta ukaz, da ostranite zaklepe in naredite delovno kopijo spet uporabno.” Upoštevajte, da v tem kontekstu *zaklep* pomeni zaklepanje datotečnega sistema in ne zaklepanje skladišča.
Delovna kopija	To je vaš krajeni “peskovnik”, področje, kjer lahko delate z datotekami pod nadzorom in se običajno nahaja na vašem trdem disku. Z uporabno ukaza “Prevzemi” naredite delovno kopijo iz skladišča, spremembe, ki jih naredite, pa shranite v skladišče z ukazom “Objavi”.
Dnevnik	Prikaže zgodovino revizij datoteke ali mape. Uporablja se tudi termin “Zgodovina”.
Dodaj	Ukaz Subversion, ki se uporablja za dodajanje datoteke ali mape v delovno kopijo. Novi elementi so dodani v skladišče, ko naredite objavo.
FSFS	Ekskluziven datotečni sistem za skladišča sistema Subversion. Uporablja se lahko tudi v omrežnih datotekah v skupni rabi. Privzet sistem pri različicah 1.2 in novejših.
GPO	Group policy object.
Izbriši	Ko izbrišete datoteko pod nadzorom (in objavite spremembo), tega elementa ni več v skladišču od objavljene revizije naprej. Seveda pa še vedno obstaja v prejšnjih revizijah skladišča in tako lahko še vedno dostopate do njega. Če je potrebno, lahko skopirate izbrisan element in ga pripeljete nazaj med žive s celotno zgodovino.
Izvozi	Ta ukaz ustvari kopijo mape pod nadzorom, prav takšno, kot je delovna kopija, vendar brez krajevnih map `.svn`.
Kopiraj	V skladišču Subversion lahko naredite kopijo datoteke ali celotnega drevesa. Kopije so izvedene kot “poceni kopije”, kar je nekaj podobnega kot povezava na originalen element in ne zavzema skoraj nič prostora. Kopija ohrani zgodovino originala, tako da lahko vedno pogledate spremembe, ki so bile narejene preden se je element prekopiral.
Lastnost	Poleg vodenja različic map in datotek Subversion omogoča tudi vodenje različic meta podatkov - rečemo jim “lastnosti” - za vsak element pod nadzorom. Vsaka lastnost ima ime in vrednost, podobno kot register. Subversion interno uporablja nekaj posebnih lastnosti, n. pr. `svn:eol-style`. Tudi TortoiseSVN uporablja nekaj internih lastnosti, n. pr. `tsvn:logminsize`. Lahko pa dodate svoje lastnosti s poljubnim imenom in vrednostjo.
Lastnost revizije (revprop)	Prav tako, kot imajo lastnosti datoteke, jih imajo tudi revizije v skladišču. Nekatere posebne se dodajo samodejno, ko se revizija ustvari: `svn:date svn:author svn:log`, ki predstavljajo datum/čas objave, avtorja in sporočilo dnevniškega zapisa. Vrednosti teh lastnosti lahko urejate, vendar upoštevajte, da niso pod nadzorom različic, tako da so spremembe trajne in se jih ne da povrniti.
Objavi	Ta ukaz sistema Subversion se uporablja za pošiljanje sprememb v delovni kopiji nazaj na skladišče. S tem se tam ustvari nova revizija.

Okrivi	Ta ukaz se uporablja samo pri besedilnih datotekah. Vsako vrstico datoteke označi z informacijo, v kateri reviziji je bila nazadnje spremenjena in kdo je spremembo naredil. Grafični ekvivalent temu ukazu je TortoiseBlame, ki vam pokaže tudi datum in dnevniški zapis, če greste z miško na številko revizije.
Popravek	Če so se v delovni kopiji spremenile le besedilne datoteke, je možno z ukazom Razlikuj ustvariti datoteko s povzetkom sprememb v obliki poenotene razlike (unified diff). Datoteka takšnega tipa se pogosto imenuje “popravek” in se lahko uporablja za pošiljanje sprememb po elektronski pošti vašemu sodelavcu ali dopisnemu seznamu. Ta datoteka se potem uporabi na neki drugi delovni kopiji. Nekdo, ki nima pravic pisanja po skladišču, lahko naredi spremembe in ustvari popravek, ki ga objavi nek drug uporabnik, ki ustrezne pravice ima. Ali pa če niste prepričani, da so vaše spremembe dobre, lahko na ta način drugim uporabnikom pošljete spremembe v pregled.
Posodobi	Ta ukaz prenese vse zadnje spremembe iz skladišča v delovno kopijo, pri tem pa spoji spremebe, ki so jih naredili ostali uporabniki, z vašimi spremembami v delovni kopiji.
Povrni	Subversion hrani krajevno “izvirno” kopijo vsake datoteke, kakršna je bila ob zadnji posodobitvi. Če ste na datoteki naredili spremembe in jih želite povrniti, uporabite ukaz “povrni”, da dobite nazaj izvirno datoteko.
Preklop	Prav tako kot ukaz “posodobi-na-revizijo” spremeni časovno okno delovne kopije, da izgleda tako, kot v neki določeni točki v zgodovini, ukaz “Preklopi” spremeni prostorsko okno delovne kopije, da kaže na drugo mesto v skladišču. To je uporabno predvsem takrat, kot delate na glavni veji in na stranskih vejah, na katerih je spremenjenih le nekaj datotek. Delovno kopijo lahko preklapljate, prenesejo pa se le spremenjene datoteke.
Premakni	Če se skladišče premakne (ker ste ga premaknili v drugo mapo na strežniku ali pa se je spremenilo domensko ime), morate narediti “premik” delovne kopije, da bodo poti URL do skladišča kazale na novo lokacijo. Opomba: ta ukaz uporabljajte le, če delovna kopija kaže na isto lokacijo znotraj istega skladišča, skladišče pa se je premaknilo na drugo lokacijo. V vseh ostalih primerih uporabite ukaz “Preklopi”.
Prevzemi	Ukaz, ki ustvari krajevno delovno kopijo v prazni mapi in prenese datoteke pod nadzorom iz skladišča.
Razlikuj	Bližnica za “Prikaži razlike”. Zelo uporabno, ko želite vedeti, kaj točno se je spremenilo.
Reši	Ko so datoteke v delovni kopiji sporne (zaradi spajanja), mora uporabnik te spore rešiti z uporabo urejevalnika (ali pa programa TortoiseMerge). Procesu rečemo “reševanje sporov”. Ko ste spore rešili, označite datoteko kot rešeno, kar vam omogoča njeno objavo.
Revizija	Vsakič ko objavite spremembe, naredite novo “revizijo” v skladišču. Vsaka revizija predstavlja stanje skladišča ob določeni točki v njegovi zgodovini. Če želite, si lahko ogledate skladišče, kakršno je bilo ob reviziji številka N. Drug pogled: revizija je množica sprememb, ki so bile narejene, ko je bila revizija ustvarjena.
Revizija BASE	Trenutna osnovna revizija datoteke v vaši *delovni kopiji*. To je različica datoteke, kakršna je bila pri zadnjem prevzemu, posodobitvi ali objavi. Revizija BASE običajno ni ista kot revizija HEAD.
Revizija HEAD	Najnovejša revizija datoteke ali mape v *skladišču*.

Skladišče	Skladišče je središčna lokacija, kjer se podatki shranjujejo in vzdržujejo. Skladišče je lahko lokacija, kjer so shranjene številne baze podatkov ali datoteke za distibucijo preko omrežja, lahko pa je tudi lokacija, ki je neposredno dostopna uporabniku, ne da bi uporabljal omrežje.
Spoji	Proces dodajanja sprememb iz skladišča v vašo delovno kopijo, ne da ob tem povozite krajevne spremembe. Včasih sprememb ni možno samodejno spojiti in takrat je delovna kopija sporna. Spajanje se zgodi samodejno, ko posodabljate delovno kopijo. Lahko pa opravljate spajanje specifičnih sprememb iz druge veje z uporabo ukaza Spoji.
Spor	Ko spremembe iz skladišča spojimo s krajevnimi spremembami, se včasih spremembe zgodijo na isti vrstici. V takšnem primeru se Subversion ne more sam odločiti, katero različico naj uporabi in datoteka je v spornem stanju. Pred objavo morate datoteko ročno urediti in rešiti spor.
SVN	Pogosto uporabljena okrajšava za Subversion. Ime lastnega protokola sistema Subversion, ki ga uporablja strežnik skladišč “svnserve”.
Uvoz	Ukaz sistema Subversion, ki omogoča uvoz celotne hierarhije map v skladišče v eni reviziji.
Veja	Ta izraz, ki se pri nadzoru različic pogosto uporablja, opisuje, kaj se zgodi, ko se razvoj prelomi v določeni točki in se nadaljuje v dveh smereh. Iz glavne veje lahko ustvarite vejo, da razvijate novo zmožnost programa, ne da bi naredili glavno vejo nestabilno. Lahko pa naredite vejo iz stabilne verzije programa, da naredite popravke, razvoj pa se nadaljuje na nestabilni veji. V sistemu Subversion so veje izvedene kot “poceni kopije”.
Zaklep	Ko element pod nadzorom zaklenete, ga s tem označite v skladišču in preprečite objave, razen iz delovne kopije, ki je zaklep zahtevala.
Zgodovina	Prikaže zgodovino revizij datoteke ali mape. Uporablja se tudi termin “Dnevnik”.

Stvarno kazalo

Simboli

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Z